Käyttäjien seuraaminen verkossa

Toni Helenius

Tampereen yliopisto
Luonnontieteiden tiedekunta
Tietojenkäsittelytieteiden tutkinto-ohjelma
Pro gradu -tutkielma
Ohjaaja: Jyrki Nummenmaa
21.11.2018

Tampereen yliopisto
Luonnontieteiden tiedekunta
Tietojenkäsittelytieteiden tutkinto-ohjelma
Toni Helenius: Käyttäjien seuraaminen verkossa
Pro gradu -tutkielma, 63 sivua
Marraskuu 2018

Tiivistelmä

Tutkielma käy läpi erilaisia tapoja, joilla käyttäjää seurataan verkossa myös sivustojen välillä paljastaen käyttäjän selaushistoriaa. Seurantaa harjoitetaan käyttäjän mieltymyksien, ostoshistorian tai muun henkilökohtaisen tiedon selvittämiseksi esimerkiksi mainostajien, käytettävyystutkijoiden tai maksunvälittäjien toimesta.

Käyttäjän selaimeen asetettavien tallenteiden avulla käyttäjään voidaan kiinnittää tunniste, jolla käyttäjä tunnistetaan verkkopalvelussa toistuvilla käynneillä. Muun muassa kolmannen osapuolen evästeillä käyttäjää voidaan tunnisteen avulla seurata myös sivustojen välillä paljastaen käyttäjän selaushistoriaa. Kolmannen osapuolen sisällöntuottajan suosiosta riippuen käyttäjän seuraaminen voi kattaa laajaakin osaa verkosta, ja sisällöntuottajat voivat edelleen tehdä yhteistyötä keskenään käyttäjien tietojen ja selaushistorian jakamiseksi toistensa kanssa.

Seurantaa on mahdollista harjoittaa myös hyödyntämällä yksilöivää tietoa käyttäjän selaimesta, käyttöjärjestelmästä ja laitteesta. Internetin toiminnallisuuksien ja sitä käyttävien laitteiden ja ohjelmistojen monipuolisuudesta johtuen erilaisia kokoonpanoja on monia, samalla mahdollistaen käyttäjän mahdollisesti tarkankin rajauksen muista käyttäjistä. Mitä enemmän käyttäjien välillä vaihtelevaa tietoa esimerkiksi ohjelmistoversioista ja laitteista saadaan kerättyä, sitä paremmin käyttäjä saadaan rajattua muista käyttäjistä. Kokoonpanosta kerätyllä tiedolla eli selaimen sormenjälkitunnisteella käyttäjä pyritään tunnistamaan toistuvilla käyntikerroilla, tai jopa sivustojen välillä, sillä sormenjälkitunniste pysyy samana eri sivustoilla.

Tutkielmassa esitellään myös keinoja, joilla seurantaa voi pyrkiä torjumaan. Selaimen asetuksilla ja lisäosilla käyttäjä voi vaikuttaa tallenteiden elinaikaan ja sivustojen väliseen näkyvyyteen. Sormenjälkitunnistaminen ja sen torjuminen on kehittyvä tutkimuksen alainen aihe, ja siihen pyritään vaikuttamaan yhdenmukaistamalla käyttäjien selaimen sormenjälkitunnisteita. Lisäosilla tai asetuksilla

voidaan pyrkiä esittämään käyttäjän kokoonpano samanlaisena kuin suurella joukolla muuta käyttäjiä, vähentäen käyttäjän yksilöitävyyttä.

Termiluettelo

AJAX Asynchronous JavaScript And XML

CDN Content distribution network

CSRF Cross-site request forgery

DNS Domain Name System

DNT Do Not Track

GPS Global Positioning System

HTTP Hypertext Transfer Protocol

HTTPS HTTP over Transport Layer Security

IP Internet Protocol

TCP Transmission Control Protocol

TCP/IP Transmission Control Protocol / Internet Protocol

TLS Transport Layer Security

URL Uniform Resource Locator

WebGL Web Graphics Library

WebRTC Web Real-Time Communication

XHR XMLHttpRequest

XSS Cross-site scripting

Sisällys

1 Johdanto

Verkon käyttäjän selaushistoria on monella tapaa hyvin arvokas. Käyttäjälle itselleen se on arvokas, sillä se saattaa sisältää paljon arkaluonteista tietoa liittyen muun muassa hänen ostoshistoriaan, terveydentilaan, kiinnostuksen kohteisiin ja varallisuuteen. Mainostajalle se tarjoaa mahdollisuuden profiloida käyttäjä ja vaikkapa selvittää, mitä tuotteita hän on vailla. Selaushistorian avulla työnantaja voi valvoa alaisiaan, tai viranomainen kansalaisia.

Käyttäjän tunnistamiseen käytetään useimmiten tietokoneelle asetettavia tallenteita, esimerkiksi evästeitä. Käyttäjän koneelle tallennetaan esimerkiksi käyttäjän asiakastiliin tai istuntoon liittyvä tunniste, joka voidaan jatkossa yhdistää käyttäjään. Tunnistaminen voi olla tarpeen verkkopalvelun toiminnallisuuksien, kuten kirjautumisen tai ostoskorin toteuttamiseen, tai sitä voidaan hyödyntää käyttäjän seuraamiseen verkossa. Seuraaminen tapahtuu usein yhtä sivustoa käyttäessä pitkäaikaisesti, tai useamman sivuston välillä käyttäjän selatessa verkkoa.

Selainvalmistajat ovat pyrkineet helpottamaan tallenteiden hallintaa ja poistoa, sekä tarjoamaan yksityisen selauksen tilaa, jonka aikana käytettävien tallenteiden elinikä rajoitetaan istunnon aikaiseksi. Selaimet ovat kuitenkin oletusasetuksilla edelleen hyvin alttiita käyttäjän seuraamiselle, ja mainostajat hyödyntävät tätä selvittääkseen käyttäjän selaushistoriaa. Selainten toiminnallisuuksien avulla on myös löydetty keinoja verkkopalveluiden toimesta vaikeuttaa käyttäjää tunnistavien tallenteiden poistoa. Esimerkiksi selaimen sormenjälkitunnistamisen avulla tallenteiden käyttö voidaan mahdollisesti ohittaa kokonaan, jolloin käyttäjän seuraaminen on täysin läpinäkyvää ja hänen tietämättömissä. Sormenjälkitunnistamisessa verkkopalvelu kerää käyttäjän laitteen ja selaimen kokoonpanosta ja asetuksista niin paljon tietoa, että hänet voidaan mahdollisesti yksilöidä muiden käyttäjien joukosta.

Luvussa 3 kerron yleisesti käyttäjää verkossa seuraavista tahoista. Tässä tutkielmassa keskityn verkkopalvelun ensimmäisen ja kolmannen osapuolen seurantamenetelmiin, jotka esittelen lyhyesti luvussa 4. Luku 5 keskittyy tilallisiin menetelmiin, joissa tietoa, esimerkiksi yksilöllinen käyttäjätunniste, talletetaan käyttäjän tietokoneelle. Luvussa 6 kerron tilattomista menetelmistä, joissa käyttäjän kokoonpanosta pyritään keräämään tietoa käyttäjän yksilöimiseksi. Viimeisessä luvussa 7 pohdin mitä vastakeinoja käyttäjällä on tutkielmassa esitettyjen seurantamenetelmien estämiseksi tai niiden tehokkuuden vähentämiseksi.

2 Tutkielmassa käytettäviä termejä

Tutkielmassa esitetään useita Internetistä, verkkopalveluista ja verkossa tapahtuvasta seurannasta puhuttaessa käytettävissä olevia termejä. Termit ovat peräisin muun muassa WWW-standardien, kuten HTTP:n ja HTML:n suunnitteludokumenteista, sekä Internetissä tapahtuvaa seurantaa käsittelevistä julkaisuista.

Verkon käyttäjä käyttää Internetin verkkopalveluita esimerkiksi tietokoneella, kännykällä tai muulla Internet-selaimella varustetulla laitteella. Selain kommunikoi verkkopalveluita tarjoavien palvelinten kanssa useimmiten käyttämällä HTTP-protokollaa, jonka avulla käyttäjän selain esimerkiksi pyytää palvelinta lähettämään verkkopalvelun sisältöä, tai syöttää tietoja verkkopalveluun. Selain esittää palvelimelta saadut tiedot käyttäjän luettavissa olevassa muodossa.

Verkkopalveluiden interaktiivisia toiminnallisuuksia mahdollistaa muun muassa niiden käytössä olevat tallenteet sekä asiakaspuolella ajettava ohjelmakoodi. Tallenteiden, esimerkiksi selaimen evästeiden, avulla tietokoneelle voidaan tallentaa käyttäjän verkkopalveluun liittyvää tietoa, joka voi olla vaikkapa asiakastilille kirjautuminen tai verkkokaupan ostoskoriin lisätyt tuotteet. Käyttäjän tietoja hänen vieraillessa verkkopalvelussa kutsutaan myös nimellä istunto.

Verkkopalvelussa voi olla käytössä sisältöä myös ulkopuolisesta, eri domain-osoitteessa palvelevasta, verkkopalvelusta. Ulkopuolista verkkopalvelua kutsutaan tällöin kolmanneksi osapuoleksi, ja sen tarjoamaa sisältöä kolmannen osapuolen sisällöksi. Kolmannen osapuolen sisällön toiminnallisuuksissa on eroja käyttäjän vierailema verkkopalvelun sisältöön eli ensimmäisen osapuolen sisältöön. Muun muassa kolmannen osapuolen evästeet eroavat merkittävästi ensimmäisen osapuolen evästeistä käyttäjän seurantaan mahdollistavien ominaisuuksien osalta.

Toisaalta interaktiivisuutta toteuttavat ominaisuudet mahdollistavat myös käyttäjän seuraamisen verkossa. Käyttäjän tallenteisiin talletettua yksilöllistä merkkijonoa, eli käyttäjän tunnistetta, vertaamalla käyttäjä voidaan todentaa samaksi useammilla käyntikerroilla. Ohjelmakoodi puolestaan mahdollistaa käyttäjän laitteiston, selaimen ja asetusten selvittämisen, ja näiden yhdistelmä supistaa samaa yhdistelmää käyttävien käyttäjien joukkoa, parhaimmillaan yksilöiden käyttäjän muiden joukosta. Laitteiston, selaimen ja asetusten yhdistelmän, eli selaimen sormenjälkitunnisteen, tehokkuutta mitataan muun muassa entropialla.

3 Käyttäjää verkossa uhkaavat tahot

Verkon käyttäjällä on monia "vastustajia" (adversary), jotka pystyvät halutessaan seuraamaan käyttäjän toimia verkossa. Esimerkiksi käyttäjän oma Internet-palveluntarjoaja (ISP) näkee kaikki DNS-pyynnöt, jotka selain lähettää palveluntarjoajan DNS- eli nimipalveluun. Käyttäjillä on useimmiten oletuksena käytössä palveluntarjoajan nimipalvelu (Upturn, 2016). DNS-pyynnöllä selain selvittää verkkopalvelun domain-osoitteeseen liittyvän palvelimen `IP`-osoitteen. Vasta kun `IP`-osoite on selvitetty voi käyttäjän selain lähettää kutsun palvelimelle. Palveluntarjoaja saa siis usein selville kaikkien verkkopalveluiden domain-osoitteet, joilla käyttäjä vierailee, eli joiden `IP`-osoitteita hän pyytää nimipalvelusta (Upturn, 2016).

Internet-palveluntarjoaja näkee salaamattomat HTTP-pyyntöjen ja palvelimen vastausten sisällöt, jotka käyttäjä tekee salaamattomilla sivuilla. Tämä johtuu siitä, että salaamaton verkkoliikenne siirtyy käyttäjältä verkkoon ja takaisin palveluntarjoajan kautta selkotekstinä. Palveluntarjoaja voi lukemisen lisäksi muokata käyttäjän verkkoliikennettä lisäämällä esimerkiksi omia mainoksiaan (Thomas et al., 2015). Yksityisyyden vaarana on myös, että hakupyynnöt ja -vastaukset voivat sivuston sisällön lisäksi sisältää esimerkiksi käyttäjän lomakkeisiin syöttämiään tietoja. Siksi useimmat selaimet pyrkivät varoittamaan, jos käyttäjä on aikeissa kirjautua tai täyttää esimerkiksi luottokorttitietojansa verkkosivulle, jota ei ole salattu HTTPS-protokollalla (Google, 2016). HTTPS-protokolla estää 'välikäsiä', esimerkiksi Internet-palveluntarjoajaa, lukemasta käyttäjälle arkaluontoisia tietoja salaamalla verkkoliikenteen sisällön. Googlen Chrome-selain alkaa versiosta 68 alkaen ilmoittamaan kaikki salaamattomat verkkosivustot tekstillä "Ei turvallinen", riippumatta siitä onko sivustolla lomakkeita (Google, 2018).

Huolimattomasti turvattujen verkkopalveluiden käyttäjät voivat altistua hakkerien tekemille hyökkäyksille. Tällaisia ovat esimerkiksi XSS- ja CSRF-hyökkäykset. XSS-hyökkäyksessä hakkeri asettaa verkkopalveluun omaa ilkeämielistä Javascript-ohjelmakoodia, jolla hän pyrkii muokkaamaan käyttäjän tietoja palvelussa tai saamaan niitä omiin käsiinsä (OWASP, 2006b). Ilkeämielinen ohjelmakoodi voi esimerkiksi lukea käyttäjän asiakastiliin liittyvää tietoa verkkopalvelusta ja lähettää sitä ulkopuoliselle palvelimelle. Muun muassa verkkopalvelun julkista kommenttiosiota voidaan käyttää ohjelmakoodin alustana, mikäli sitä ei ole turvattu tällaisen hyökkäyksen varalta. Myös mainostajaverkkoja voidaan käyttää ilkeämielisen ohjelmakoodin levitykseen (Sood ja Enbody, 2011).

CSRF-hyökkäyksessä puolestaan käyttäjä päätyy hakkerin ylläpitämälle verkkosivulle, joka komentaa käyttäjän selaimen lähettämään toiseen verkkopalveluun hakupyynnön. Jotkin verkkopalvelut eivät tarkista, että tuleeko käyttäjän lähettämä kutsu heidän oman verkkopalvelun kautta, eli käyttäjän omasta tahdosta, vai ulkopuoliselta verkkosivulta hakkerin kirjoittaman ohjelmakoodin toimesta. Käyttäjän ollessa kirjautuneena verkkopalveluun, heikosti turvattu palvelu hyväksyy kutsun käyttäjän lähettämänä, ja näin hakkeri voi muokata tai lukea käyttäjän nimissä verkkopalvelusta tietoa. (OWASP, 2006a)

Yksi suurista yksityisyyden uhista verkon selaajille on selainten lisäosat. Lisäosat voivat seurata käyttäjän toimia esimerkiksi lisäämällä selattaville sivustoille JavaScript-koodia, joka tallentaa sivulta, lomakkeista tai hakulausekkeista tietoa ja lähettää ne ulkopuolisille. Jagpal et al. (2015) havaitsivat 2012-2015 välisenä aikana lähes joka kymmenennen Chrome Web Storeen ladatun lisäosan olevan ilkeämielisiä.

Selaimeen esimerkiksi Chrome Web Storesta tai Firefox Add-ons -sivustolta asennetut lisäosat ovat myös oletuksena automaattisesti päivittyviä. Käyttäjän selain siis hakee automaattisesti kehittäjän julkaisemat lisäosaan tehdyt muutokset. On tullut ilmi tapauksia, joissa lisäosan kehittäjä on myynyt lisäosan eteenpäin, ja uusi omistaja on muuttanut lisäosan ilkeämieliseksi lisäämällä esimerkiksi mainoksia tai haittaohjelmia. Koska lisäosien automaattinen päivittyminen on usein käyttäjälle läpinäkyvää, niiden muuttumista ilkeämieliseksi on vaikea huomata. (Kapravelos et al., 2014)

4 Käyttäjän seurantamenetelmät

Tässä tutkielmassa käsittelen ensimmäisen ja kolmannen osapuolen käytettävissä olevia seurantamenetelmiä. Ensimmäisellä osapuolella tarkoitetaan verkkopalvelua itseään, toisella osapuolella käyttäjää ja kolmannella osapuolella verkkopalvelun käyttämiä ulkopuolisia, toisessa domain-osoitteessa toimivia, tahoja (Järvinen, 2010). Kolmannen osapuolen palveluilla pyritään usein tarjoamaan käyttäjälle lisäsisältöä, esimerkiksi kuvia tai upotettuja videoita, harjoittamaan tiedonkeräystä lisäämällä analytiikkaa, veloittamaan maksuja tuotteista tai palveluista ulkopuolisella maksunvälittäjällä. Kolmannen osapuolen palvelut toimivat eri domain-osoitteesta kuin itse verkkopalvelu.

Ensimmäisen osapuolen seurannassa verkkopalvelu pyrkii seuraamaan käyttäjiä ja heidän toimiaan palvelussa. Käyttäjien tietokoneelle voidaan asettaa esimerkiksi yksilöivä käyttäjätunniste, jonka avulla heidät voidaan tunnistaa myös useamman käyntikerran välillä. Käyttäjän pitkäaikaisella seurannalla voidaan pyrkiä esimerkiksi selvittämään hänen mieltymyksiään tai verkkopalvelun käyttötapoja. Ensimmäisen osapuolen seurannassa käyttäjän selaushistoria verkkopalvelun ulkopuolella voi jäädä suurimmilta osin piiloon, elleivät verkkopalvelut tee yhteistyötä keskenään (Sorensen, 2013).

Kolmannen osapuolen seurannassa verkkopalvelu käyttää ulkopuolista tahoa esimerkiksi sisällön tuottamiseen tai toiminnallisuuksien toteuttamiseen. Käyttäjä on silloin yhteydessä kolmannen osapuolen kanssa verkkopalvelun välityksellä tämän lisättyä kolmannen osapuolen resursseja sivustolle. Siinä missä XSS- ja CSRF-hyökkäykset ovat täysin ulkopuolisen hakkerin toteuttamia luvattomia toimia, antaa verkkopalvelu kolmannen osapuolen palveluja käyttäessään sille valtuudet käyttäjältä saataviin tietoihin (Mayer ja Mitchell, 2012). Kolmas osapuoli voi esimerkiksi käyttäjän ladatessa kuvaa kerätä tietoa käyttäjän hakupyynnöstä ja tallentaa tietoa käyttäjän tietokoneelle, jota hyödyntää seuraavilla latauskerroilla. Jotkin kolmannen osapuolen resurssit, esimerkiksi analytiikkaan liittyvät palvelut, voivat myös kerätä tietoa käyttäjän toimista vierailemassaan verkkopalvelussa.

Mayer ja Mitchell (2012) jakoivat ensimmäisen ja kolmannen osapuolen harjoittamat käyttäjän seurantamanetelmät kahteen eri tyyppiin: tilallisiin ja tilattomiin seurantamenetelmiin. Tilallisissa menetelmissä käyttäjän koneelle talletetaan tietoja, joista käyttäjä tunnistetaan seuraavilla käyntikerroilla. Tilattomissa menetelmissä käyttäjän selaimesta lasketaan yksilöivä sormenjälkitunniste, joka pysyy samana käyttäjän selatessa verkkoa.

Käyttäjä voi olla mahdollista tunnistaa verkkopalvelussa myös hänen käyttäytymisen perusteella. Weiss et al. (2007) tutkivat hiiren liikuttamisen yksilöitävyyttä käyttäjien välillä. Deutschmann ja Lindholm (2013) seurasivat käyttäjän näppäimistön, hiiren ja ohjelmien käyttöä istunnon turvaamiseksi keräämällä ja vertaamalla käyttäjän tiedoista kerättyä profiilia sen hetkiseen käyttöön. Käyttäytymisprofiileihin perustuvat seurantamenetelmät sivuutetaan tässä tutkielmassa.

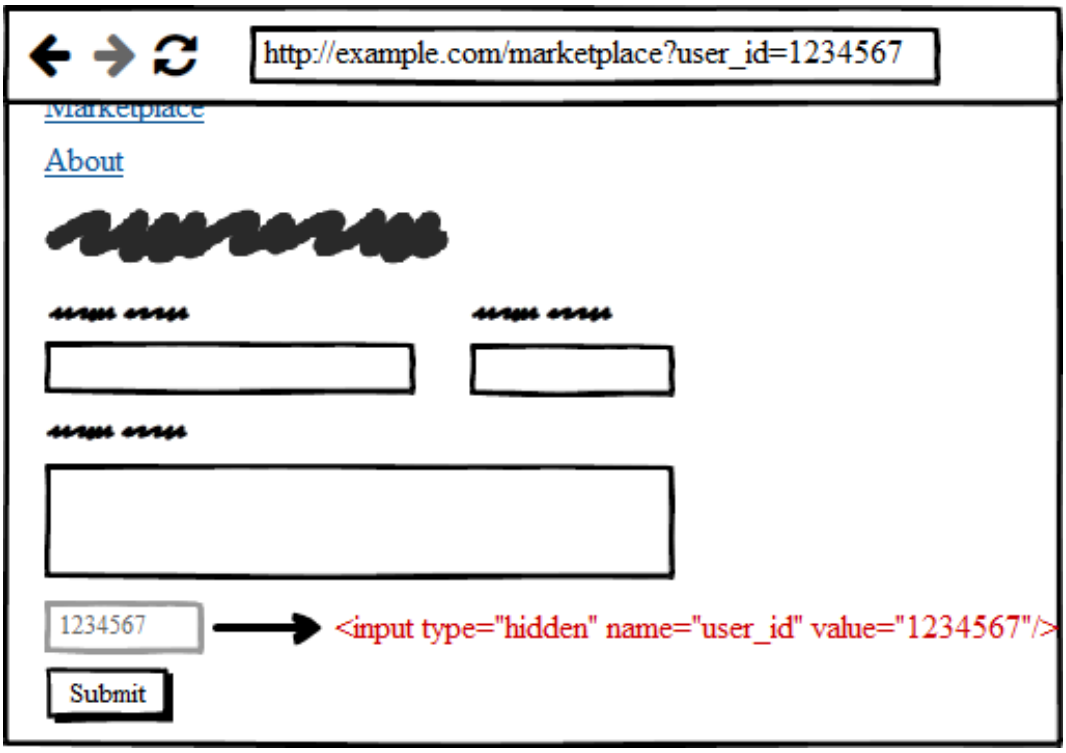
5 Tilalliset käyttäjän seurantamenetelmät

Tiedon välitykseen verkkopalveluiden ja käyttäjän välillä käytetään HTTP-protokollaa. HTTP on tilaton (stateless) protokolla, joten käyttäjän hakupyyntöjen välillä ei ole yhteyttä keskenään verkkosivustoa käyttäessä (Järvinen, 2010). Käyttäjän tila pitää siis jotenkin sisällyttää pyyntöjen ja vastausten mukana kuljetettavaksi, ja tähän voidaan käyttää esimerkiksi piilotettuja lomakekenttiä tai sivuston URL-osoitteen yhteydessä käytettäviä hakulausekkeita (query string). Yleisin keino on käyttää evästeitä, jotka voivat olla istunnon aikaisia tai pysyviä.

Istunnon aikaisesti tunnistaessa käyttäjän tila säilytetään vain tilapäisesti. Yksi käyttötapa tilapäiseen tilan säilyttämiseen on esimerkiksi verkkokaupan ostoskorin toteuttaminen. Vaikka HTTP-hakupyyntöjen välillä ei ole yhteyttä, voi verkkokauppa tällä tavoin muistaa, mitkä tuotteet asiakas on lisännyt koriin hänen jatkaessa tuotteiden selaamista (Järvinen, 2010). Käyttäjän jättäessä verkkokaupan selain unohtaa tilapäiset tallenteet, ja samalla ostoskori tyhjentyy. Verkkopalveluun palatessaan palvelin ei enää tiedä käyttäjästä mitään, ellei käyttäjä jotenkin tunnistaudu uudelleen (Silvennoinen, 2004).

Jos käyttäjä halutaan tunnistaa pysyvästi, käyttäjän tietokoneelle voidaan tallentaa yksilöivä tunniste, käyttämällä esimerkiksi pysyvää evästettä. Pysyvien evästeiden avulla käyttäjä voidaan tunnistaa käyntikertojen välillä, sillä ne ovat istunnon sijaan voimassa ennalta asetettavaan päivämäärään asti. Käyttötapa tähän on esimerkiksi verkkokaupan asiakkaan kirjautumisen muistaminen. Käyttäjän palatessa verkkokauppaan se voi tunnistaa asiakkaan ja hänen aikaisemmin antamat tiedot ja ostokset, ja voi esimerkiksi tervehtiä asiakasta hänen omalla nimellään (Järvinen, 2010). Kun evästeet on kerran asetettu, selain pyrkii säilyttämään ne asetettuun päivämäärään asti, ellei käyttäjä itse siihen jotenkin vaikuta esimerkiksi tyhjentämällä evästeet selaimestaan (Mozilla, 2014c).

Samalla pysyvät keinot säilyttää käyttäjän tila tuovat mukanaan monia yksityisyyteen liittyviä kysymyksiä mahdollistaessaan käyttäjän pitkäaikaisen seuraamisen, sillä tunnisteet jäävät tietokoneelle ja niiden poistaminen käyttäjän vastuulle. Jo vuonna 1999 New York Times julkaisi artikkelin evästeiden käytöstä Internetin valvontatyökaluna, sillä pitkäaikaisen käyttäjätunnisteen avulla käyntikerrat voidaan yhdistää toisiinsa (Acar et al., 2014). Jos verkkopalvelut ovat jonkinlaisessa yhteistyössä keskenään tai käyttävät yhteistä kolmannen osapuolen palvelua, voi käyttäjää seurata useamman eri verkkopalvelun välillä (Acar et al., 2014).



Kuva 5.1: Piilotettuun lomakekenttään on upotettu käyttäjän tunniste. Se on näkymätön käyttäjälle ja lähtee takaisin palvelimelle käyttäjän täytettyä lomakkeen.

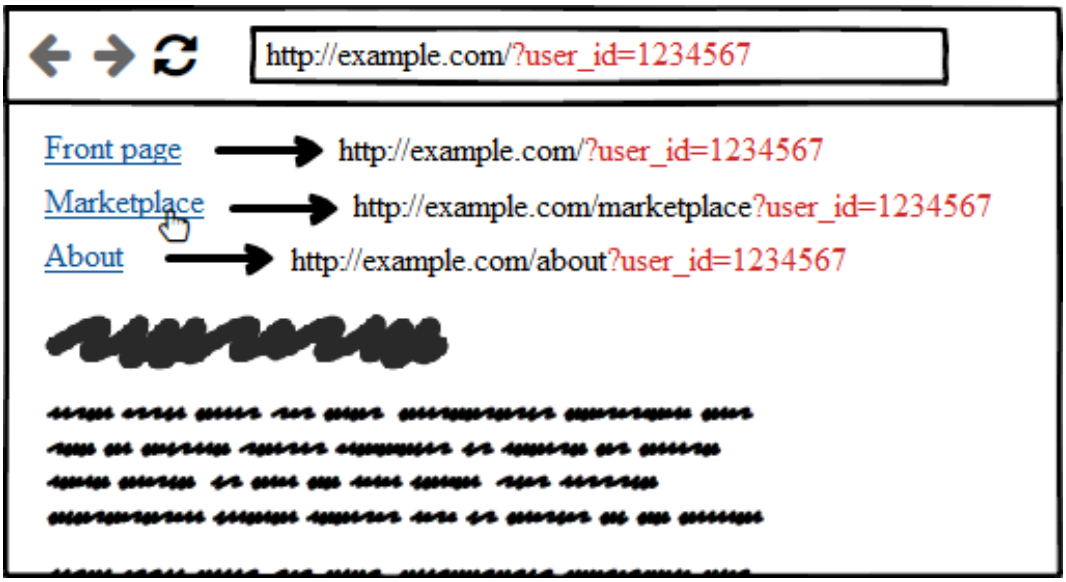
5.1 Istunnon aikainen tunnistaminen

Istunnon aikaisessa tunnistamisessa käytetään keinoja, jotka pääasiassa säilyvät vain sivuston käyntikerran ajan tai selaimen aukioloajan.

Piilotettuja lomakekenttiä käytettäessä käyttäjän tila voidaan sisällyttää sivua ladatessa valmiiksi lomakkeeseen, jonka käyttäjä täyttää. Piilotettuihin lomakekenttiin voi olla talletettuna esimerkiksi käyttäjän aikaisemmin antamat tiedot, tai käyttäjän tunniste. Tällöin käyttäjän lähettäessä lomakkeen palvelin saa sekä lomakkeeseen syötetyt uudet tiedot, että piilotetuissa lomakekentissä olevat aikaisemmin annetut tiedot. (Silvennoinen, 2004) Kuvassa 5.1 on esimerkki lomakkeesta, joka sisältää piilotetun kentän käyttäjän tunnistetta varten.

Lomakekenttien käyttöä hankaloittaa se, että ne säilyvät käyttäjän ja palvelimen välisessä kierrossa vain niin kauan kuin käyttäjä lähettää lomakkeen. Esimerkiksi linkkien avulla navigoidessa lomakekentät eivät siirry mukana toiselle sivulle.

Hakulausekkeita käytettäessä käyttäjän tila välittyy hakupyyntöjen välillä si-



Kuva 5.2: Hakulausekkeessa kulkeva käyttäjätunniste näkyy sivun osoitteen perässä. Kaikki sivustolla olevat linkit sisältävät tunnisteen, jotta se välittyy käyttäjän selatessa sivustoa.

vun osoitteen perään liitettyinä parametreina. Tällöin niitä voi käyttää sivuston sisäisissä linkeissä sekä lomakkeissa mahdollistaen tilan säilyttämisen käyttäjän vuorovaikutusten välillä.

Hakulausekkeiden käyttöön liittyy kuitenkin ongelmia. Jakaessaan linkin käyttäjä saattaa huomaamattaan välittää henkilökohtaisen istuntonsa toiselle henkilölle. Kuvassa 5.2 on esimerkki hakulausekkeen avulla toteutetusta istunnosta. Jos käyttäjä kopioi sivun tai jonkin sivulla olevan linkin osoitteen jakaakseen välittää hän samalla myös istuntonsa. Toisaalta myös ilkeämielinen hakkeri voi päästä käsiksi käyttäjän tietoihin ohjaamalla hänet verkkopalveluun linkillä, joka sisältää valmiiksi asetetun hakkerin tiedossa olevan istuntotunnisteen. Käyttäjän kirjauduttua on istuntotunnisteen avulla myös hakkeri kirjautuneena palveluun käyttäjän tunnuksilla. (Schmucker, 2011)

Evästeet (cookies) ovat hakulausekkeita ja piilotettuja lomakekenttiä turvallisempi tapa tunnistaa käyttäjä (Kolšek, 2002). Palvelin asettaa evästeen käyttäjän selaimeen lähettämällä sen hakupyynnön vastauksen mukana. Evästeelle annetaan nimi, arvo ja mahdollisesti aika johon asti se on voimassa. Evästettä, johon aikarajaa ei ole asetettu, kutsutaan istunnon aikaiseksi evästeeksi (session cookie). Istunnon aikaiset evästeet poistuvat, kun käyttäjä on sulkenut selaimen, joskin jotkut selaimet voivat palauttaa ne, jos käyttäjä käyttää selaimen ominai-

suutta, jolla palauttaa aikaisempi istunto selaimen avautuessa (Mozilla, 2014c).

Istunnon aikaiset evästeet ovat kuitenkin liian tilapäisiä, että niitä voisi käyttää pitkäaikaiseen käyttäjän seurantaan.

5.2 Evästeet

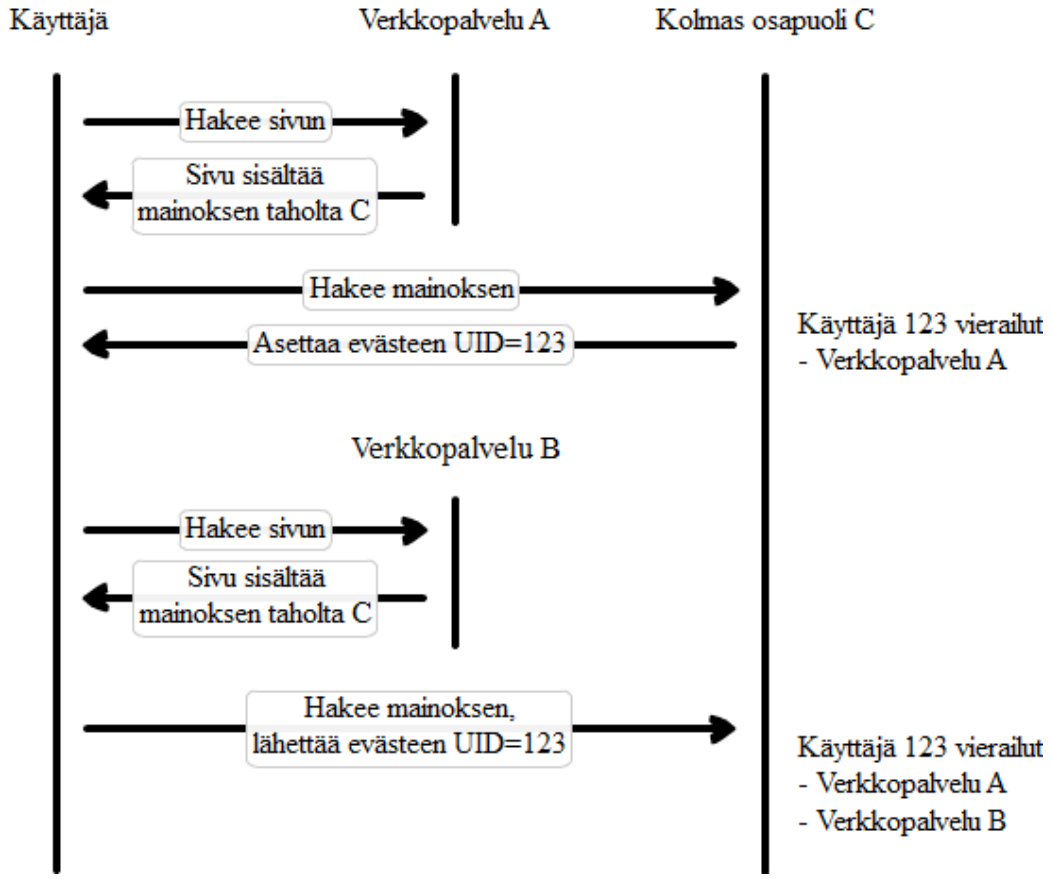
Kun evästeeseen lisätään voimassaoloaika, sitä kutsutaan pysyväksi evästeeksi (persistent cookie) (Mozilla, 2014c). Tällöin käyttäjän tilan hallinta ei jää vain yhden istunnon aikaiseksi, vaan sitä voi käyttää myös myöhemmillä käyntikerroilla. Selain pyrkii säilyttämään pysyvät evästeet voimassaoloaikaan asti, ellei käyttäjä itse aikaisemmin poista niitä. Näin niillä voidaan esimerkiksi tunnistaa käyttäjä useiden selainistuntojen ajan, ilman että käyttäjän tarvitsee uudestaan kirjautua verkkopalveluun (Järvinen, 2010). Evästeet toimivat laajimmillaan yhteen domain-osoitteeseen liitettynä, tai niiden toimivuutta voi edelleen rajoittaa alidomain-osoitteeseen tai hakemistopolkuun (Mozilla, 2014c).

Kun käyttäjä vierailee sivustolla, jossa hänellä on voimassa oleva aikaisemmalla käyntikerralla asetettu eväste, lähetetään eväste `HTTP`-hakupyynnön mukana takaisin palvelimelle. Siksi sitä voidaan käyttää erinomaisesti käyttäjän tunnistamiseen. Käyttäjän ei myöskään tarvitse välttämättä olla kirjautunut verkkopalveluun tai omistaa tunnuksia siihen, vaan hänelle voidaan asettaa esimerkiksi satunnainen merkkijono, jolla verkkopalvelu yksilöi käyttäjän (Broenink, 2012). Vastaanotettuaan evästeen palvelin voi edelleen muuttaa sitä esimerkiksi lisäämällä tietoa tai vaihtamalla voimassaoloaikaa pidemmälle tulevaisuuteen. Yksilöllisen käyttäjätunnisteen avulla verkkopalvelu osaa yhdistää käyttäjän vierailut palvelussa toisiinsa, ja sen avulla lisätä toiminnallisuutta tai kerätä informaatiota käyttäjän toimista (Järvinen, 2010).

Evästeet ovat yleisin ja yksinkertaisin tapa toteuttaa käyttäjän tunnistaminen toistuvilla käyntikerroilla verkkopalveluun. Kun eväste asetetaan sivuston ulkopuolisesta domain-osoitteesta, puhutaan kolmannen osapuolen evästeistä. Niiden avulla käyttäjä voidaan mahdollisesti tunnistaa useamman sivuston välillä.

5.3 Kolmannen osapuolen evästeet

Verkkopalvelu voi sisältää ulkopuolisista verkkopalveluista liitettyä sisältöä, joka voi olla esimerkiksi kuvia, skriptejä tai vaikkapa videota. Tällaista eri domain-osoitteesta sisältöään tarjoavaa ulkopuolista tahoa kutsutaan tässä tapauksessa kolmanneksi osapuoleksi (Järvinen, 2010). Kolmansia osapuolia voi olla useam-



Kuva 5.3: Kolmannen osapuolen evästeen asettaminen ulkopuolisen tahon C toimesta. Eväste sisältää käyttäjän yksilöivän käyttäjätunnisteen, joka välitetään myöhemmin taholle C käyttäjän vieraillessa toisessa verkkopalvelussa. Taho C tietää lopuksi käyttäjän vierailleen kahdessa eri verkkopalvelussa.

pia, kun sisältöä on liitettynä useammasta eri domain-osoitteesta. Niitä käytetään usein esimerkiksi mainosten tarjoamiseen, sisällön jakeluun (CDN), yhteisöpalvelujen integraatioon tai analytiikkaan (Mayer ja Mitchell, 2012). Toisella osapuolella tarkoitetaan verkkopalvelun käyttäjää ja ensimmäisellä osapuolella käyttäjän vierailemaa verkkopalvelua (Järvinen, 2010).

Myös kolmannen osapuolen tahot voivat asettaa evästeitä. Tällöin niitä kutsutaan kolmannen osapuolen evästeiksi (third-party cookies). Kolmannen osapuolen evästeitä ovat sellaiset evästeet, jotka asetetaan verkkopalvelun ulkopuolisesta domain-osoitteesta. Evästeet ovat laajimmillaan luettavissa yhdestä domain-osoitteesta, joten ensimmäinen osapuoli ei pääse käsiksi kolmannen osapuolen evästeisiin. Toisaalta kolmas osapuoli voi saada käyttäjän vierailuista tietoa useammas-

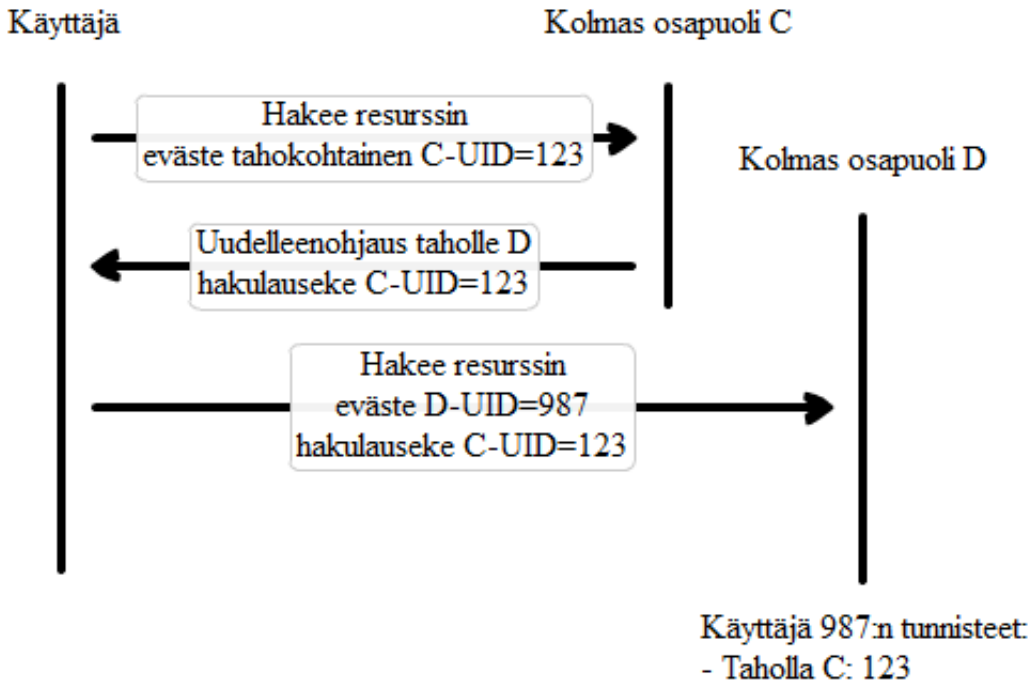
sa verkkopalvelussa. Tämä toimii siksi, että samat kolmannen osapuolen evästeet ovat käytössä kaikissa verkkopalveluissa, jossa kolmannen osapuolen sisältöä käytetään (Mozilla, 2014c). Evästeiden avulla kolmas osapuoli, vaikkapa mainostentarjoaja, voi siis mahdollisesti saada selville paljon kattavampaa selaushistoriaa kuin ensimmäinen osapuoli, eli verkkopalvelu itse.

Tieto kolmannelle osapuolelle käyttäjän vierailemasta sivustosta välittyy esimerkiksi resurssin osoitteen loppuun liitettävällä hakulausekkeella tai `Referer`-otsakkeella. Ensimmäisessä tapauksessa resurssia kolmannelta osapuolelta hakiessa voidaan `URL`-osoitteen loppuun liittää hakulauseke, johon merkitään tietoa käyttäjän tällä hetkellä vierailemasta sivustosta. `Referer`-otsake on selainten itse `HTTP`-kutsuun lisäämä otsake, joka ilmoittaa kutsun alkuperän, eli sivuston osoitteen, joka pyytää resurssia. Kolmannen osapuolen palvelusta upotetut skriptit voivat ilmoittaa käyttäjän tietoa myös `AJAX`-kutsuin. Skriptit pääsevät käsiksi upotustavasta riippuen vähintäänkin omaan evästeeseensä sekä käyttäjän sivuston osoitteeseen ja sivun 'title'-otsikkoon. (Krishnamurthy et al., 2011)

Kolmannen osapuolen evästeet välitetään aina evästeet asettaneelle taholle, kun sisältöä ladataan kyseisen kolmannen osapuolen domain-osoitteesta. Mikäli kaksi eri verkkopalvelua käyttää samaa kolmannen osapuolen tahoa sisällön tuottamiseen, käytetään siis molemmissa verkkopalveluissa vieraillessa samaa evästettä, kun sisältöä ladataan samalta kolmannen osapuolen taholta. Kuvassa 5.3 on kuvattuna tapaus, jossa vieraillessaan verkkopalvelussa A palvelu lataa käyttäjälle mainoskuvan ulkopuoliselta taholta C, joka asettaa samalla evästeen käyttäjän selaimeen. Kun käyttäjä edelleen vierailee verkkopalvelussa B, joka käyttää samaa mainosten tuottajaa, lähettää selain aikaisemmin asetetun evästeen ulkopuoliselle taholle. Kuvatussa tapauksessa evästeen sisältönä on yksilöivä käyttäjätunniste, jolloin kolmannen osapuolen taho saa tietää käyttäjän vierailleen molemmissa palveluissa.

5.3.1 Evästeiden synkronointi

Kolmannen osapuolen sisällöntuottaja saa siis tiedon käyttäjän liikkeistä kaikissa verkkopalveluissa, joissa sen tarjoamat palvelut ovat käytössä. Muun muassa mainostajat voivat käyttää kattavampaa selaushistoriaa käyttäjien profiloimiseen, jonka avulla mainokset voidaan kohdistaa tarkemmin esimerkiksi demografian, tarpeiden tai mielenkiintojen avulla. Käyttäjän selailtua tuotteita verkkokaupassa voidaan vastaavia suositella mainoksin hänelle muualla verkossa sisällöntuottajan toimesta. Kolmannen osapuolen sisällöntuottajat voivat edelleen laajen-



Kuva 5.4: Käyttäjä vierailee verkkosivustolla, jolla on sisältöä kolmannen osapuolen taholta C, ja tämä on sopinut tietojen jakamisesta tahon D kanssa. Käyttäjän selain hakee resurssia taholta C, jolloin tämä lukee käyttäjälle antamansa tunnisteen evästeestä. Hakupyyntö uudelleenohjataan taholle D, samalla lisäten tunniste mukaan hakulausekkeena. Taho D saa hakupyynnön vastaanottaessaan luettua sekä oman käyttäjälle asettamansa tunnisteen että tahon C käyttäjälle antaman tunnisteen hakulausekkeesta.

taa kattavuuttaan sopimalla tietojen jakamisesta keskenään (Castelluccia et al., 2014). Kattavammalla profiililla käyttäjän profiilia ja mainoksien tehokkuutta voidaan edelleen parantaa.

Evästeiden domain-osoitekohtaisuuden takia niitä ei pysty lukemaan kuin ne asettanut taho itse. Halutessaan kolmannen osapuolen tahot voivat kuitenkin käyttää esimerkiksi evästeiden synkronointia (‘cookie syncing’ tai ‘cookie matching’) niiden jakamiseen keskenään. Evästeet synkronoimalla pyritään käyttäjistä kerätyt tiedot tekemään yhdistettäviksi toisiinsa. Kun kumpikin taho osaa yhdistää omat käyttäjätunnisteet toisen tahon käyttäjätunnisteisiin, voidaan esimerkiksi käyttäjältä talletetut selaushistoriat jakaa tahojen kesken ja yhdistää laajemmaksi selaushistoriaksi. Kuvassa 5.4 on kuvattu käyttäjätunnisteen tiedottaminen taholta C taholle D. (Castelluccia et al., 2014)

5.4 Etag

Verkossa on paljon staattista sisältöä, joiden lataaminen vaatii aikaa ja kaistaa riippuen palvelimen etäisyydestä ja resurssin koosta. Selaimissa on siksi monipuolisilla ominaisuuksilla varustettu välimuisti, johon niitä voi tallettaa valmiiksi seuraavia hakukertoja varten. Kun käyttäjä vierailee uudestaan sivustolla, jossa on jokin aikaisemmin ladattu resurssi, voi selain hyödyntää välimuistia resurssin uudelleen lataamisen sijaan. (Mozilla, 2016)

Verkkopalvelulla on kaksi eri tapaa hyödyntää selaimen välimuistia, jotka toimivat `HTTP`-otsakkeilla `Cache-Control` ja `ETag`. `Cache-Control`-otsakkeella voidaan asettaa ladatulle resurssille voimassaoloaika, johon asti selain voi ohittaa `HTTP`-hakupyynnön tekemisen kokonaan ja tarjota resurssin välimuistista. `ETag`-otsakkeella puolestaan resurssille voidaan asettaa tunniste, jolla selain voi kysyä palvelimelta, onko aikaisemmin ladatusta resurssista uudempaa versiota. Yleinen tapa tunnisteen luomiseen on ottamalla resurssin tiiviste (hash). Jos välimuistiin ladatussa resurssissa on `Cache-Control`-voimassaoloaika päättynyt, lähettää selain `HTTP`-hakupyynnön mukana tunnisteen `If-None-Match`-otsakkeessa. Palvelin vertaa tällöin tämänhetkisen resurssin tunnistetta käyttäjältä saatuun tunnisteeseen, ja niiden osuessa vastaa tilakoodilla `304 Not Modified` ilman sisältöä. Resurssia ei siis ladata uudestaan, vaan selain esittää sen sijaan välimuistiin aikaisemmin tallennetun resurssin. (Mozilla, 2015a)

`ETag`-tunnisteelle ei ole kuitenkaan asetettu ennalta määrättyä tapaa, jolla se luodaan (IETF, 2014). Vapaasti valittavissa olevaa `ETag`-tunnistetta voidaan käyttää siksi myös käyttäjien tunnistamiseen, jolloin sitä kutsutaan välimuistievästeeksi (Cache cookie) (Juels et al., 2006). Verkkopalvelu voi siis asettaa ensimmäisellä käyntikerralla resurssiin yksilöivän käyttäjätunnisteen, jonka jälkeen tunniste lähetetään takaisin palvelimelle seuraavilla käyntikerroilla. Välimuistia käyttävät `ETag`-tunnisteet eivät myöskään katoa, kun evästeet otetaan pois käytöstä tai tyhjennetään selaimesta, jolloin käyttäjä ei välttämättä tiedä niiden mahdollistamasta seurannasta (Ayenson et al., 2011).

`ETag` on resurssikohtainen, joten se asetetaan usein kerrallaan yhteen sivustolla olevaan `URL`-osoitteella haettavana olevaan kohteeseen. Useimmiten tunnistamistarkoitukseen käytetään jotakin sivuston laajuisesti läpinäkyvää ja huomaamatonta kuvaa. Tällöin sitä voidaan käyttää sivuston laajuisesti.

Myös `ETag`iä on mahdollista käyttää sivustojen väliseen tunnistamiseen. Kun välimuistieväste asetetaan sivuston ulkopuolisella resurssilla, voidaan `HTTP`-kutsun mukana lähetettävää `Referer`-otsaketta hyödyntää selvittämään käyttäjän

sillä hetkellä vierailema verkkopalvelu. Kun toinen verkkopalvelu käyttää samaa resurssia, lähetetään `HTTP`-kutsun mukana jälleen tunniste ja sivuston osoite `ETag`- ja `Referer`-otsakkeissa, samaan tapaan kuin kolmannen osapuolen evästeitä käyttäessä kuvassa 5.3.

5.5 Web Storage

Siinä missä evästeet ja välimuistievästeet toimivat `HTTP`-kutsujen yhteydessä, on myös asiakaspuolella ajettavilla skripteillä mahdollista tallentaa tietoa käyttäjän koneelle. Evästeiden lisäämisen ja muokkaamisen lisäksi JavaScript-ohjelmilla on käytettävissä HTML5-standardin mukana tuomat tiedon tallentamisen ominaisuudet kuten Web Storage. Käyttäjän tunnistamiseen käytettyjä asiakaspuolen tallenteita kutsutaan HTML5-evästeiksi (Ayenson et al., 2011).

Web Storage standardiin kuuluu kaksi asiakaspuolen tallennusmekanismia: Session Storage ja Local Storage. Ne ovat JavaScript-ohjelmointirajapinnalla toimivia avain-arvo-varastoja. Session Storage säilyttää tiedot istunnon ajan nimensä mukaisesti, ja Local Storagea voi käyttää tiedon pitkäaikaiseen säilyttämiseen. Evästeistä poiketen Local Storageen tallennettaviin tietoihin ei aseteta päättymispäivämäärää vaan ne säilytetään siihen asti, kun palvelu tai käyttäjä itse tyhjentää ne (Ayenson et al., 2011). Web Storageen talletetut tiedot ovat alkuperään sidottuja evästeiden tavoin, joten niihin pääsee käsiksi vain verkkopalvelussa ajettavilla ensimmäisen tai kolmannen osapuolen asettamilla asiakaspuolen ohjelmilla (Mozilla, 2014e).

5.6 IndexedDB

Selaimet tarjoavat skriptien käyttöön myös selaimessa pyöriviä tietokantoja, joista tuetuin on tällä hetkellä IndexedDB. Nämä toimivat samaan tapaan kuin Local Storage, eli niihin voi säilyttää tietoa, kunnes palvelu tai käyttäjä tyhjentää ne (Mozilla, 2014a). Selaimesta ja tallenteiden tyhjennystavasta riippuen ne voivat kuitenkin tarjota edelleen pitkäikäisemmän tallenteen kuin evästeet tai Web Storage.

5.7 Flash-evästeet

Flash-liitännäistä voi hyödyntää myös käyttäjien seurannassa, vaikka liitännäisen käyttö onkin vähentynyt huomattavasti ja jatkaa vähenemistä (Projects, 2018). Ayenson et al. (2011) mittasivat Flash-evästeiden (Flash cookie) käyttöä sadalla

suosituimmalla sivulla ja huomasivat sen käytön vähentyneen vuosien 2009–2011 välillä huomattavasti. Acar et al. (2014) mittasivat seurantaan käytettäviä Flash-evästeitä löytyvän kymmenestä 200:sta suosituimmasta verkkopalvelusta.

Flash-evästeet tarjoavat monien muiden tallenteiden tavoin pitkäkestoisen varaston käyttäjätunnisteen tallentamiselle, joskin liitännäisen yksityisyydensuojaa on sittemmin pyritty kehittämään Adoben ja selainvalmistajien toimesta (Adobe, 2011). Flash-evästeet ovat kuitenkin oletuksena selaimesta riippumattomia eli ne saattavat toimia jopa useamman selaimen välillä (Boda et al., 2012).

5.8 Zombievästeet

Yen et al. (2012) tutkivat evästeiden elinikään vaikuttavia tekijöitä ("cookie churn") mittaamalla Bing-hakukonetta käyttävien käyttäjien evästeitä. Kuukauden mittaisia aikajaksoja tarkastellessa keskimäärin 45 prosenttia evästeistä ei havaittu ensimmäisen päivän jälkeisen kuukauden aikana uudelleen. Syynä he arvelivat palvelun käytön lopettamisen lisäksi käyttäjien aktiivisen evästeiden tyhjentämisen, yksityisen selauksen käytön, sekä selainasetukset, jotka tyhjentävät evästeet esimerkiksi selaimen sammutuksen yhteydessä.

Evästeiden lyhytikäisyydestä johtuen muun muassa mainostajat ja tutkijat ovat pohtineet evästeen eliniän pidentämistä hyödyntämällä myös selaimen muita tallennusmekanismeja. Evästeitä, jotka palauttavat itsensä näitä muita tallennusmekanismeja käyttäen, kutsutaan usein muun muassa nimillä Zombieväste (Zombie cookie) ja Supercookie (Sorensen, 2013). Zombievästeiden tehokkuus perustuu siihen, että niissä hyödynnettävät tallennusmekanismit tyhjentyvät tai tyhjennetään usein eri tavoilla toisistaan. Esimerkiksi selaimien manuaalinen evästeiden tyhjennys ei useimmiten vaikuta välimuistiin ja tällöin välimuistievästeisiin. Jos käyttäjä tyhjentää siis vain evästeet, voi verkkopalvelu palauttaa tunnisteen evästeisiin selaimen välimuistia hyödyntämällä.

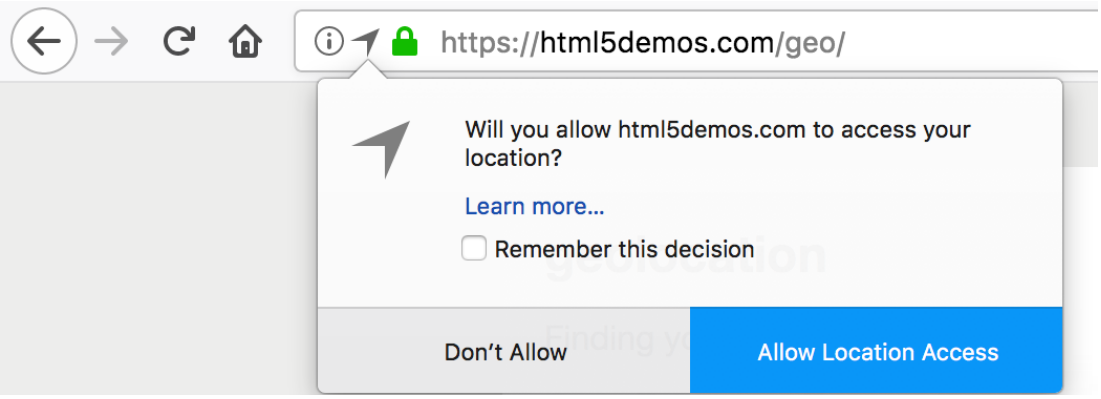
Listauksessa 5.1 on esimerkki tavallisen evästeen ja välimuistievästeen yhteiskäytöstä. Esimerkissä käyttäjän tunnisteena on hänen lempinimensä ja se löytyy evästeestä ‘nickname’. Välimuistievästeenä käytettävä `ETag` lähetetään käyttäjän `HTTP`-kutsun mukana `If-None-Match`-otsakkeessa. Sivu pyrkii säilyttämään käyttäjän tunnisteen sekä evästeissä, että välimuistissa, jotta jos toinen on poistettu tai poistunut tunniste saadaan vielä mahdollisesti palautettua. Välimuistievästettä käytetään yleensä jonkin sivulla olevan kuvan kanssa, sillä `ETag`it ovat resurssikohtaisia. Tällöin samaa välimuistievästettä voi käyttää koko sivun laajuisesti tavallisen evästeen tapaan. Yksinkertaisuuden vuoksi tässä esimerkissä kuitenkin

```
/**
 * HTTP endpoint for /cookie-refreshing page.
 * @param req HTTP request object
 * @param res HTTP response object
 */
function cookieRefreshing(req, res) {
    const cookie = req.cookies["nickname"]; // Get nickname from cookie
    const etag = req.get("If-None-Match"); // Get nickname from ETag header
    if (cookie || etag) {
        res.cookie("nickname", cookie || etag, {
            path: "/cookie-refreshing",
            expires: new Date(Date.now() + 1000 * 60 * 60 * 24)
        }); // Refresh cookie.
        res.set("ETag", cookie || etag); // Refresh ETag.
    }
    const response = cookieRefreshingHTML(
        cookie || etag || "Unknown",
        cookie || "...",
        etag || "..."
    );
    res.send(response);
}
```

Listaus 5.1: Yksinkertainen zombievästetoteutus Express-verkkosovelluskehyksellä. Eväste ja `ETag` päivitetään, jos käyttäjän tunniste löytyy edes toisesta `HTTP`-otsakkeista. Koko verkkosivun toteutus löytyy liitteestä A.

tunniste on liitettynä `ETag`inä suoraan itse sivuun.

Zombievästeissä käytetään yleensä kaikkia mahdollisia käyttäjän selaimesta löytyviä tallennusominaisuuksia. Sama tunniste talletetaan mahdollisuuksien mukaan muun muassa evästeisiin, välimuistiin, Web Storageen, Web-tietokantoihin kuten IndexedDB:en ja Flash-evästeisiin. Useissa selaimissa evästeiden tyhjennyksen mukana poistuvat kaikki tallenteet paitsi välimuisti. Silti monissa selaimissa, ja varsinkin vanhentuneissa selainversioissa, on paljon puutteellisia ominaisuuksia ja oletusasetuksia, jotka tekevät Zombievästeiden käyttämisestä edelleen tehokasta (Sorensen, 2013). Myös selainten tarjoamissa rajapinnoissa on puutteellisuuksia, jotka hankaloittavat käyttäjän jäljitystä estävien selainlisäosien tekemistä.



Kuva 6.1: Firefox-selaimen valintaruutu hyväksynnän vaativan Geolocation rajapinnan käyttöön.

6 Tilattomat seurantamenetelmät

Tilattomissa seurantamenetelmissä käyttäjän tietokoneelle ei tallenneta mitään tunnistetta. Sen sijaan käyttäjän selaimesta, käyttöjärjestelmästä, laitteesta ja käyttöympäristöstä pyritään keräämään niin paljon tietoa, että käyttäjä voidaan joko yksilöidä tai rajata mahdollisimman pieneen joukkoon käyttäjiä, jotka jakavat saman laite- ja ohjelmistokokoonpanon ja asetukset (Acar et al., 2013). Talteen otettua yksilöivää tietoa kutsutaan niin sanotuksi sormenjälkitunnisteeksi. Itse menetelmää kutsutaan julkaisusta riippuen selaimen tai laitteen sormenjälkitunnistamiseksi, sillä tietoa kerätään usein molemmista lähteistä.

Tietoa voidaan kerätä palvelimelle saapuvien `HTTP`-kutsujen lisäksi asiakaspuolella selaimessa ajettavilla ohjelmilla. `HTTP`-kutsut sisältävät tietoa muun muassa käyttäjän selaimen ja käyttöjärjestelmän versioista ja kielestä. Asiakaspuolella ajettavilla JavaScript- ja Flash-ohjelmilla tietoa saadaan esimerkiksi näytön resoluutiosta, asennetuista fonteista ja liitännäisistä sekä käyttäjän aikavyöhykkeestä. Tiedot, jotka vaihtelevat eri käyttäjien selainten välillä, sopivat sormenjälkitunnistamiseen ja rajaavat heitä yhä pienempiin joukkoihin. (Eckersley, 2010)

Toisaalta sormenjälkitunnisteeseen kerättyjen tietojen halutaan usein olevan aikaa myöden vakaita, jotta sormenjälkitunniste pysyisi samana jokaisella käyntikerralla myös muutosten, kuten selaimen päivityksen, jälkeen. Sormenjälkitunnisteen voidaan myös haluta pysyvän samana myös yksityisen selauksen tilassa. Siksi sormenjälkitunnistamisessa käytettäviä ominaisuuksia pyritään arvioimaan niiden vakauden perusteella. (Nikiforakis et al., 2015)

Tutkielmaan valitut tilattomat seurantamenetelmät ovat sellaisia, jotka ei-

vät useimmilla selaimilla vaadi erillistä hyväksyntää käyttäjällä, ja joiden käyttö on siten käyttäjän vaikeasti havaittavissa. Esimerkiksi Geolocation rajapinnalla voidaan paikantaa käyttäjä `GPS`:ää, tukiasemapaikannusta tai WiFi-paikannusta käyttäen. Käyttäjän tulee hyväksyä rajapinnan käyttö kuvan 6.1 näkyvän valintaruudun mukaisesti, ennen kuin se on verkkopalvelun käytettävissä. Tällaiset hyväksynnän vaativat toiminnallisuudet ovat tässä tutkielmassa sivuutettu.

6.1 Entropia

Tilattomien seurantamenetelmien tehokkuutta verrataan tieteellisissä julkaisuissa usein entropialla. Entropia mittaa tiedon sisällön ennalta-arvaamattomuutta ja se kasvaa muuttujan mahdollisten arvojen vaihtelevuuden lisääntyessä. Muuttujan entropia H lasketaan summaamalla sen mahdollisten arvojen informaatiosisältöjen ja todennäköisyysten tulot (Eckersley, 2010):

$$H(X) = \sum_{i=1}^{n} P(X_i)I(X_i), \tag{6.1}$$

missä $P(X_i)$ on muuttujan jonkin arvon todennäköisyys ja $I(X_i)$ sen informaatiosisältö. Muuttujan informaatiosisältö I määritellään

$$I(X) = \log_2(\frac{1}{P(X_i)}). \tag{6.2}$$

Logaritmin kantalukuna on 2, sillä alan julkaisuissa entropiaa mitataan useimmiten bitteinä. Entropian määritelmän voi edelleen kirjoittaa auki Eckersleyn (2010) käyttämään muotoon

$$H(X) = -\sum_{i=1}^{n} P(X_i)\log_2(P(X_i)). \tag{6.3}$$

Muuttuja X on selaimen sormenjälkitunnisteen tehokkuutta laskiessa jokin käyttäjän selaimen ominaisuus. Jos mitattavana ominaisuutena on esimerkiksi `HTTP`-kutsun `User-Agent` -otsakkeesta haettava käyttöjärjestelmäversio, sen mahdollisia arvoja ovat esimerkiksi ‘Windows 10’, ‘MacOS Sierra’ tai ‘Ubuntu 16.04’. Mitä ennalta-arvaamattomampi käyttäjän käyttöjärjestelmäversio on, sitä suurempi on kyseisen ominaisuuden entropia.

Muuttujista kerättävät entropiat voidaan myös summata, jolloin saadaan niiden yhdistelmästä muodostuva entropia. Keräämällä tietoa useammasta ominaisuudesta niistä muodostuu yhä harvinaisempia yhdistelmiä entropian kasvaessa. Toisaalta selaimen sormenjälkitunnisteen tehokkuutta laskiessa monet muuttujat

ovat sidoksissa toisiinsa. Esimerkiksi käyttöjärjestelmää ‘MacOS Sierra’ käyttävillä on todennäköisemmin käytössä ‘Safari’ selain kuin ‘Windows 10’ käyttäjillä. Jotkin ominaisuuksien yhdistelmät ovat siis yleisempiä kuin toiset.

Entropian bittien voidaan nähdä puolittavan sormenjälkitunnisteeseen samaistuvat käyttäjät. Jokainen lisäbitti tarkentaa sormenjälkitunnistetta ja keskimäärin jakaa saman sormenjälkitunnisteen omaavien käyttäjien (anonymiteettijoukko, anonymity set) lukumäärän kahtia (Eckersley, 2010). Tällöin voidaan ajatella, että 40 000 käyttäjän verkkopalvelussa 8 bitin entropian sormenjälkitunniste jakaisi käyttäjät keskimäärin $\frac{40000}{2^8} \approx 156$ käyttäjän joukkoihin.

6.1.1 Normalisoitu entropia

Koska ominaisuuksista tai sormenjälkitunnisteista lasketut entropiat riippuvat tutkimuksen aineiston koosta eli kerättyjen sormenjälkitunnisteiden määrästä, eivät ne suoraan ole vertailukelpoisia julkaisujen välillä. Joissain julkaisuissa käytetään entropian lisäksi Laperdrix et al. (2016) esittämää normalisoitua entropiaa, mikä ottaa huomioon myös otoskoon. Normalisoitu entropia määritellään

$$NH = \frac{H(X)}{H_M}, \tag{6.4}$$

jossa H_M tarkoittaa pahinta tapausta, jossa jokainen muuttujan arvo on yksilöllinen, ja samalla yhtä todennäköinen. Tällöin entropia on suurin mahdollinen. Kaava voidaan edelleen kirjoittaa auki muotoon

$$NH = \frac{H(X)}{\log_2(N)}. \tag{6.5}$$

Kuvassa 6.2 on listattuna joitakin sormenjälkitunnisteissa käytettäviä ominaisuuksia ja niiden lasketut entropiat. Normalisoitu entropia on laskettu jakamalla entropia alimmalla rivillä olevan otoskoon eli kerättyjen sormenjälkitunnisteiden määrän logaritmilla.

6.2 Passiivinen sormenjälkitunnistus

Sormenjälkitunnistaminen jaetaan usein kahteen ryhmään: aktiiviseen sormenjälkitunnistamiseen sekä passiiviseen sormenjälkitunnistamiseen. Passiivisessa sormenjälkitunnistamisessa käyttäjä pyritään tunnistamaan `HTTP`-kutsujen avulla ilman asiakkaan laitteella ajettavia skriptejä. Tämä on käyttäjälle täysin näkymätöntä, sillä tietojen keruusta ei jää jälkeä käyttäjän laitteelle, eikä asiakaspuolelta ole löydettävissä tietoa keräämiä ohjelmia. Passiivista sormenjälkitunnista

Attribute	Panopticlick		AmIUnique		Dataset		Mobile devices		Desktop/laptop machines	
	Entropy	Norm.	Entropy	Norm.	Entropy	Norm.	Entropy	Norm.	Entropy	Norm.
Platform	-	-	2.310	0.137	1.200	0.057	2.274	0.127	0.489	0.024
Do Not Track	-	-	0.944	0.056	1.919	0.091	1.102	0.061	1.922	0.092
Timezone	3.040	0.161	3.338	0.198	0.164	0.008	0.551	0.031	0.096	0.005
List of plugins	15.400	0.817	11.060	0.656	9.485	0.452	0.206	0.011	10.281	0.494
Use of local/session storage	-	-	0.405	0.024	0.043	0.002	0.056	0.003	0.042	0.002
Use of an ad blocker	-	-	0.995	0.059	0.045	0.002	0.067	0.004	0.042	0.002
WebGL Vendor	-	-	2.141	0.127	2.282	0.109	2.423	0.135	1.820	0.088
WebGL Renderer	-	-	3.406	0.202	5.541	0.264	4.172	0.233	5.278	0.254
Available fonts	13.900	0.738	8.379	0.497	6.904	0.329	2.192	0.122	6.967	0.335
Canvas	-	-	8.278	0.491	8.546	0.407	7.930	0.442	8.043	0.387
Header Accept	-	-	1.383	0.082	0.729	0.035	0.111	0.006	0.776	0.037
Content encoding	-	-	1.534	0.091	0.382	0.018	1.168	0.065	0.153	0.007
Content language	-	-	5.918	0.351	2.716	0.129	2.291	0.128	2.559	0.123
User-agent	10.000	0.531	9.779	0.580	7.150	0.341	8.740	0.487	6.323	0.304
Screen resolution	4.830	0.256	4.889	0.290	4.847	0.231	3.603	0.201	4.437	0.213
List of HTTP headers	-	-	4.198	0.249	1.783	0.085	1.941	0.108	1.521	0.073
Cookies enabled	0.353	0.019	0.253	0.015	0.000	0.000	0.000	0.000	0.000	0.000
H_M (worst scenario)	18.843		16.860		20.980		17.938		20.793	
Number of FPs	470,161		118,934		2,067,942		251,166		1,816,776	

Kuva 6.2: Panopticlick (Eckersley, 2010), AmIUnique (Laperdrix et al., 2016) ja Gómez-Boix et al. (2018) sormenjälkitunnistamista varten kerättyjen ominaisuuksien entropiat (sarake 'Entropy') ja normalisoidut entropiat (sarake 'Norm.'). Sarake 'Dataset' viittaa Gómez-Boix et al. saamiin tuloksiin, ja 'Mobile devices' ja 'Desktop/laptop computers' edelleen mobiililaitteisiin ja tietokoneisiin.

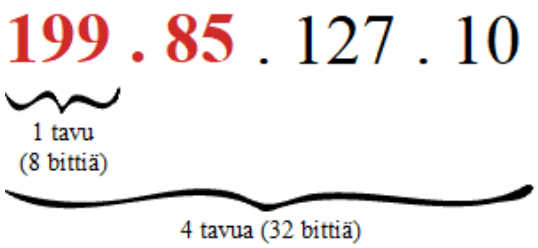
voidaan kutsua myös palvelinpuolen sormenjälkitunnistamiseksi, sillä tiedot kerätään käyttäjän ottaessa yhteyden palvelimeen `HTTP`-kutsua tehdessä. (Gulyás et al., 2008)

Passiivisessa sormenjälkitunnisteessa käytetään muun muassa `HTTP`-kutsua ja sen otsakkeita, käyttäjän `IP`-osoitetta ja `TCP/IP`-yhteyden ominaisuuksia.

6.3 IP-osoite

`IP`-osoite on käyttäjän Internet-palveluntarjoajaltaan laitteelleen saama julkinen osoite, jonka avulla tietoa voidaan reitittää laitteelle. `IP`-osoite välittyy verkkopalvelulle selaimen ottaessa yhteyden palvelimeen. Koska osoite voi vaihtua käyttäjän verkkosovittimen ottaessa uuden osoitteen käyttöönsä, käyttäjän vaihtaessa verkkoa esimerkiksi kotiverkosta työpaikan verkkoon, tai mobiililaitteen siirtyessä langattoman verkon alueelta toiselle, ei osoitetta usein käytetä yksin tunnistamaan käyttäjää. Laitteet voivat myöskin olla `NAT`-verkon tai välityspalvelimen kautta käytössä, jolloin yhden osoitteen jakaa mahdollisesti useampi käyttäjä. (Schmucker, 2011)

`IP`-osoitteen vaihtumista voi kuitenkin jonkin verran ennakoida, ainakin tapauksissa, joissa käyttäjä ei kokonaan vaihda eri Internet-palveluntarjoajan verkkoon. Sormenjälkitunnistamisessa käytetään usein osoitteen kahta tai kolmea en-



Kuva 6.3: Esimerkkiosoitteen rakentuminen neljästä oktetista, jossa kaksi ensimmäistä oktettia on merkitty paksulla tekstillä.

simmäistä tavua, jotka pysyvät samoina pidemmän aikaa (Alaca ja van Oorschot, 2016; Boda et al., 2012). `IP`-osoitteen rakentuminen tavuista on kuvattuna kuvassa 6.3.

`IP`-osoitteen voi usein kohdistaa myös maantieteellisesti. Palvelut kuten MaxMind (2002) mahdollistavat osoitteiden kohdistamisen monesti jopa kaupungin tarkkuudella. MaxMind ylläpitää palveluntarjoajien käyttämistä `IP`-osoitejoukoista ja sijainneista 'GeoIP'-tietokantaa.

6.4 `HTTP`-kutsun otsakkeet

6.4.1 `User-Agent`

Käyttöjärjestelmän, selaimen ja niiden versiot on mahdollista selvittää `User-Agent`-otsakkeesta. Otsakkeen alkuperäinen tarkoitus oli antaa verkkopalvelun hyödyntää käyttäjän käyttöjärjestelmän ja selaimen toiminnallisuuksia parhaan mahdollisen käyttökokemuksen tarjoamiseksi. HTML5-standardien kehittymisen myötä käyttöjärjestelmä- ja selainkohtaisien toiminnallisuuksien käyttäminen on kuitenkin vähenemään päin. (Nikiforakis et al., 2013)

Otsakkeesta saatavan yksilöivän tiedon määrä on runsas käyttöjärjestelmien ja selaimien versioiden lukuisista yhdistelmistä johtuen. Toisaalta eri yhdistelmien käytön osuudet voivat vaihdella yleisön mukaan, esimerkiksi maantieteellisesti. (Gómez-Boix et al., 2018)

6.4.2 `Accepted-Languages`

`Accepted-Languages`-otsake kertoo verkkopalvelulle selaimen toivomat kielet vastaanotettavalle verkkosivulle. Kielet ovat useimmiten käyttöjärjestelmään asete-

tuista kielistä riippuvaisia. Useamman käytössä olevan kielen tapauksessa niissä on mainittuna toivottu järjestys, joista verkkopalvelu voi valita käyttäjälle parhaan mahdollisen kielen sivuston esittämiseen. (Laperdrix et al., 2016)

6.4.3 Plugins

Plugin-otsakkeen ilmoittavat selaimessa käytössä olevat selaimen liitännäiset. Tällaisia voivat olla esimerkiksi selaimen PDF-lukija, tai Java- tai Flash-liitännäinen. Otsakkeen avulla verkkopalvelu näkee mitä liitännäisiä käyttäjän selaimessa on käytettävissä, ja voi tarjota käyttäjälle esimerkiksi Javalla tehdyn sovelluksen tai videota Flash-soittimella.

Mobiililaitteilla liitännäisiä ei ole käytössä, ja tietokoneilla niiden käyttöä ollaan lopettamassa Flash-liitännäistä lukuun ottamatta (Mozilla, 2017a; Apple, 2018b). Flash-tuki on myös loppumassa vuoteen 2020 mennessä (Adobe, 2017). Selaimen liitännäiset ovat kuitenkin edelleen merkittävä tekijä sormenjälkitunnistamisessa (Alaca ja van Oorschot, 2016).

6.4.4 DNT

Useimmat selaimet sisältävät asetuksen, jonka avulla käyttäjä voi halutessaan pyytää verkon toimijoita olemaan jäljittämättä ja keräämättä tietoa käyttäjästä kohdennettua mainontaa varten. Jos käyttäjä laittaa asetuksen päälle, lähetetään se HTTP-kutsun mukana DNT-otsakkeessa. Toisaalta otsakkeen käyttämisen toimivuudesta ei ole mitään takeita, sillä se on verkon tahojen omassa päätösvallassa toimia niiden mukaan. Muun muassa Acar et al. (2014) mukaan Do Not Track asetusta kunnioittaa hyvin harva, mikä tekee siitä tehottoman tavan estää jäljittämistä.

Do Not Track asetus välittyy palvelimelle HTTP-otsakkeena, joten sitä on mahdollista käyttää passiivisessa sormenjälkitunnistamisessa. Ominaisuuden tehottomuuden takia Do Not Track asetusta on kritisoitu sormenjälkitunnistusta avittavana tekijänä (Eckersley, 2010).

6.5 TCP

Selaimen lähettäessä HTTP-kutsua verkkopalveluun se ottaa yhteyden palvelimeen käyttäen TCP/IP-protokollaa. TCP/IP on TCP- ja IP-protokollien yhdistelmä, jota käytetään kahden päätelaitteen väliseen tiedonvälitykseen Internetissä.

TCP voi paljastaa tiettyjä yksityiskohtia käyttäjästä. TCP-verkkoliikennettä seuraamalla verkkopalvelin voi nähdä esimerkiksi tietoa käyttäjän laitteen käyttöjärjestelmästä, selaimesta, mahdollisesta välityspalvelimesta, ja siitä onko käyttäjä NAT-verkon takana. Yksi tällainen TCP-liikenteen passiiviseen sormenjälkitunnistamiseen keskittyvä työkalu on 'p0f v3' (Zalewski, 2012).

6.6 Aktiivinen sormenjälkitunnistaminen

Aktiivinen sormenjälkitunnistaminen tarkoittaa asiakaspuolella ajettavien skriptien käyttämistä sormenjälkitunnistamiseen. Sormenjälkitunnistusohjelma ajetaan käyttäjän selaimessa esimerkiksi JavaScriptillä tai Flash-liitännäisellä, ja kerätyt tiedot voidaan lähettää palvelimelle XHR-kutsulla.

Siinä missä passiivinen sormenjälkitunnistaminen on rajoitettu TCP/IP-liikenteen ja HTTP-otsakkeiden tulkintaan, on aktiivisessa sormenjälkitunnistamisessa mitattavissa huomattavia määriä selaimen ja laitteen asetuksia ja toiminnallisuuksia. Toisaalta asiakaspuolella tiedon hakeminen saattaa olla käyttäjän huomattavissa, eli aktiivinen sormenjälkitunnistaminen ei ole yhtä läpinäkyvää kuin passiivinen sormenjälkitunnistaminen. (Broenink, 2012)

6.7 Fontit

Käyttöjärjestelmän ja selaimen fontit ovat yksi yksilöivimmistä ominaisuuksista sormenjälkitunnistamisessa. Verkkopalveluiden käytössä oleva joukko fontteja riippuu käyttöjärjestelmästä, selaimesta, muista asennetuista ohjelmista ja käyttäjän itse asentamista fonteista. Monet ohjelmat, esimerkiksi Microsoft Office, lisäävät käyttöjärjestelmään ylimääräisiä fontteja. Fontteja myös kehitetään jatkuvasti, joten eri versioissa käyttöjärjestelmiä ja selaimia voi olla käytössä eri versiot samasta fontista. (Fifield ja Egelman, 2015)

Sormenjälkitunnistamisessa käytetään yleisesti ainakin kolmea tapaa selvittää käyttäjää yksilöllistäviä tekijöitä fonteista. Adobe Flash -liitännäinen mahdollistaa fonttien nimien listauksen, vaikka itse liitännäisen käyttö on vähenemässä. Liitännäisen avulla saa kaikki käyttäjällä käytössä olevat fontit yksinkertaisimmin selville.

Käytössä olevien fonttien selvittäminen onnistuu myös JavaScriptin avulla, mutta siinä tapauksessa fonttien olemassaoloa verrataan ennalta määriteltyyn listaan fonteista. Vertaamalla tiettyä fonttia käyttävää tekstiä samaan tekstiin, mutta selaimen oletusfontti ollessa käytössä, nähdään onko fontti käytettävissä.

Jos fonttia ei ole käytettävissä teksti näkyy oletusfontilla, eli samankokoisena kuin verrokkifontti. Mikäli tiettyä fonttia käyttäessä tekstin koko on eri kuin oletusfonttinen teksti, tiedetään että fontti on käytettävissä. Kun sama vertailu tehdään isolle listalle ennalta määriteltyjä fontteja, mitkä sormenjälkitunnistusta harjoittava taho on koonnut, nähdään mitkä kaikki fontit löytyvät käyttäjän laitteelta. (Nikiforakis et al., 2013)

Kolmannessa tavassa käyttää fontteja sormenjälkitunnistukseen lasketaan myös tekstin koot. Kokoja ei kuitenkaan verrata oletuskokoihin selvittääkseen fontin olemassa oloa, vaan itse lasketut koot otetaan talteen sormenjälkitunnisteeseen. Fonttien koot vaihtelevat eri fonttien välillä, mutta myös fonttiversioiden välillä. Eri käyttöjärjestelmissä ja selaimissa on myös fonttiversioiden välillä eroavaisuuksia. Siksi kokojen talteen ottaminen ennalta määrätystä listasta fontteja parantaa sormenjälkitunnisteen tehokkuutta entisestään. (Fifield ja Egelman, 2015)

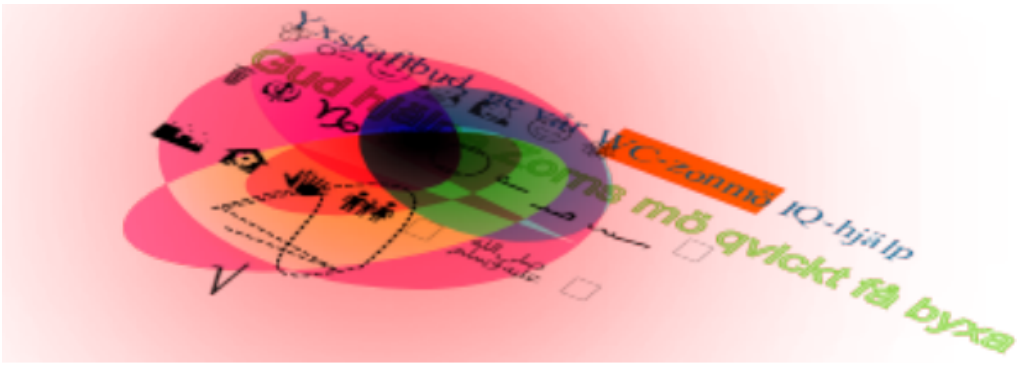
6.8 Canvas fingerprint

Canvas-elementin avulla verkkosivu voi piirtää selaimessa kaksiulotteista rasterkuvaa JavaScriptiä käyttäen. Sitä voi käyttää esimerkiksi kuvaajien, animaatioiden tai videon renderöintiin. (Mozilla, 2014b)

Samalla Canvas-elementin on kuitenkin huomattu olevan hyödyllinen sormenjälkitunnistamisessa. Sen tuottama grafiikka on riippuvainen laitteistosta, käyttöjärjestelmästä sekä selaimen toteutuksesta, joten sillä voidaan pyrkiä etsimään eroavaisuuksia eri käyttäjien välillä piirtämällä elementin avulla ennalta määritetty kuva. Piirretyn kuvan tiedot voidaan sen jälkeen ottaa talteen Canvas-rajapinnan $ToDataURL$-kutsulla, joka palauttaa kuvan värit pikseli pikseliltä. Pikselitiedosta voi sen jälkeen ottaa tiiviste, mikä toimii selaimen Canvas-sormenjälkitunnisteena (Canvas fingerprint). (Acar et al., 2014)

Canvas-sormenjälkitunnisteen yksilöivyyttä lisätään piirtämällä tekstiä. Käyttämällä selaimen oletusfontteja ja mahdollisia erikoismerkkejä kuten hymiöitä voidaan edelleen parantaa sormenjälkitunnisteen tehokkuutta. Kappaleessa 6.7 mainitut fonttiversioiden eroavaisuudet esimerkiksi käyttöjärjestelmien välillä vaikuttavat myös Canvas-sormenjälkitunnistamisessa. Kuvat 6.4 ja 6.5 ovat esimerkkejä Canvas-sormenjälkitunnistamisessa piirrettävästä kuvista, joiden pikselitiedosta otetaan tiiviste.

Kuva 6.4: Esimerkki piirrettävästä kuvasta, jota voidaan käyttää Canvas-sormenjälkitunnisteen muodostamiseen. Kuva koostuu oletus Sans Serif ja Serif-fonteilla piirretystä tekstistä, osaksi läpinäkyvästä laatikosta sekä Unicode-merkistöstä löytyvästä hymiöstä. (Laperdrix et al., 2016)



Kuva 6.5: Aiemmasta esimerkistä 6.4 jatkava Canvas-sormenjälkitunnisteessa käytettävä piirrettävä kuva. Tutkijat pyrkivät lisäämään Canvas-sormenjälkitunnisteen yksilöllisyyttä lisäämällä esimerkiksi erikoismerkkejä, rotaatioita, gradientteja ja läpinäkyvyyttä. (Gómez-Boix et al., 2018)

6.9 Web Audio

Web Audio on rajapinta äänen prosessointiin ja syntetisointiin selaimessa. Sillä voidaan toteuttaa ääniä ja ääniefektejä esimerkiksi ohjelmiin, peleihin ja esityksiin.

Englehardt ja Narayanan (2016) tekivät tutkimuksessaan katsaukseen miljoonan sivun käyttämistä seurantamenetelmistä, ja huomasivat Web Audion `Audio Context`-rajapinnan olevan käytössä sormenjälkitunnistamisessa. Menetelmä käyttää rajapinnan oskillaattoria ja kompressoria luodakseen käyttäjästä yksilöivän ääneen perustuvan sormenjälkitunnisteen. Oskillaattorilla ja kompressorilla luotu ääni vaihtelee muun muassa laitteen äänikortin ja käyttöjärjestelmän ohjelmiston mukaan, joten menetelmällä luotua äänisormenjälkitunnistetta voidaan käyttää sormenjälkitunnistamisessa (Cao et al., 2017).

6.10 WebGL

WebGL-rajapintaa käytetään 2D ja 3D-grafiikan piirtämiseen JavaScriptin avulla. Se mahdollistaa rautakiihdytetyn grafiikan ja varjostimien (shader) käyttämisen selaimessa. WebGL-rajapintaa voidaan myös hyödyntää Canvasin tapaan sormenjälkitunnistukseen, sillä sen tuottama ulostulo on riippuvainen laitteistosta ja ohjelmistosta. (Acar et al., 2014)

Cao et al. (2017) ovat lisäksi tutkineet WebGL-rajapinnan käyttöä selaintenvälisessä käyttäjän tunnistamisessa. He löysivät rajapinnasta tiettyjä selaimesta riippumattomia ominaisuuksia, joiden avulla sormenjälkitunnistetta voidaan kohdistaa käyttäjän laitteeseen yksittäisen selaimen sijaan. Tutkijat huomasivat verteksivarjostimien (vertex shaders), fragmenttivarjostimien (fragment shaders) ja läpinäkyvyyden avulla olevan mahdollista rakentaa selainriippumatonta ulostuloa, jota tulkitsemalla käyttäjiä voidaan mahdollisesti tunnistaa laitetasolla.

Näiden lisäksi he löysivät yksilöllistäviä ominaisuuksia tekstuureista, vaihtelevista muuttujista, reunanpehmennyksestä, valaistuksesta, kamerasta ja leikkelystä. Toiminnallisuuksia käyttämällä he toteuttivat listan tehtäviä, jotka käydään läpi käyttäjän selaimessa. Tehtävien tulokset muutetaan kuviksi, tiivistetään ja ovat käytettävissä sormenjälkitunnisteessa. (Cao et al., 2017)

Muita selaimesta riippumattomia löydettäviä arvoja löytyy muun muassa laitteen suorittimen ytimien lukumäärästä, näytön resoluutiosta, Web Audiosta ja käyttöjärjestelmään asennetuista kielistä. Käyttämällä näitä yhdessä WebGL-sormenjälkitunnistamisen kanssa Cao et al. (2017) rakensivat sormenjälkitunnistus-

menetelmän, jolla he saivat 83,24%:lle kävijöistä yksilöllisen sormenjälkitunnisteen ja 91,44%:ia sormenjälkitunnisteista toimimaan selaimien välillä. Toisaalta tutkimus oli pieni ja sisälsi 3 615 sormenjälkitunnistetta 1 903:lta käyttäjältä.

6.11 WebRTC

Selainten väliseen reaaliaikaiseen tiedonvälitykseen kehitettyä WebRTC:tä käytetään muun muassa videopuhelupalveluiden toteuttamiseen. Sen avulla vertaisverkon osapuolet löytävät toisensa optimaalisinta reittiä käyttäen. WebRTC:n avulla käyttäjien selaimet voivat siirtää tietoa suoraan toistensa kanssa ilman palvelimen välitystä.

WebRTC toimii sekä paikallisverkoissa että Internetissä, ja toimiakseen optimaalisesti käyttäjät välittävät palvelimen kautta toisilleen sekä paikalliset että julkiset `IP`-osoitteet. Näistä valitsemalla selaimet voivat käyttää suorinta reittiä tiedon siirtämiseen keskenään. Toisaalta palvelimen vastaanottamaa listaa `IP`-osoitteista voidaan käyttää lisänä sormenjälkitunnisteessa. Esimerkiksi `NAT`-verkon käyttäjillä voi olla sama julkinen osoite ja yksilöllinen paikallinen osoite (Englehardt ja Narayanan, 2016). Paikallisen osoitteen avulla käyttäjä saadaan yksilöityä myös `NAT`-verkon käyttäjien joukosta.

WebRTC voi myös paljastaa käyttäjän oikean `IP`-osoitteen hänen käyttäessä `VPN`-verkkoa (Alaca ja van Oorschot, 2016).

6.12 Sormenjälkitunnisteen elinikä

Selaimen sormenjälkitunniste ei ole yhtä tarkka tapa tunnistaa käyttäjä kuin käyttäjän koneelle talletettavat tunnisteet. Se muuttuu aikaa myöten laitteiston, ohjelmiston tai asetuksien vaihtuessa. Näytön vaihtuminen, selaimen päivittyminen tai uusien fonttien asentaminen tietokoneelle ovat muutamia esimerkkejä, jotka voivat johtaa selaimen sormenjälkitunnisteen muuttumiseen. Sormenjälkitunnisteessa huomioitavia ominaisuuksia valitessa pitää siis ottaa huomioon niiden muuttuminen aikaa myöten. (Vastel et al., 2018)

Taulukossa 6.6 on luokiteltuna ominaisuuksia, joita sormenjälkitunnistamisessa voidaan käyttää. Vastel et al. (2018) tutkivat ominaisuuksia ja niiden keskimääräistä elinikää. He luokittelivat ominaisuudet muutoksen aiheuttavan tekijän mukaan kolmeen luokkaan. Itsestään muuttuvat (automatic evolutions) ominaisuudet johtuvat esimerkiksi selaimen päivittymisestä. Käyttöympäristön mukaan muuttuvat (context-dependent evolutions) vaihtuvat esimerkiksi käyttäjän mat-

		Percentile (days)		
Attribute	**Trigger**	**50th**	**90th**	**95th**
Resolution	Context	Never	3.1	1.8
User agent	Automatic	39.7	13.0	8.4
Plugins	Automatic/User	44.1	12.2	8.7
Fonts	Automatic	Never	11.8	5.4
Headers	Automatic	308.0	34.1	14.9
Canvas	Automatic	290.0	35.3	17.2
Major browser version	Automatic	52.2	33.3	23.5
Timezone	Context	206.3	53.8	26.8
Renderer	Automatic	Never	81.2	30.3
Vendor	Automatic	Never	107.9	48.6
Language	User	Never	215.1	56.7
Dnt	User	Never	171.4	57.0
Encoding	Automatic	Never	106.1	60.5
Accept	Automatic	Never	163.8	109.5
Local storage	User	Never	Never	320.2
Platform	Automatic	Never	Never	Never
Cookies	User	Never	Never	Never

Kuva 6.6: Sormenjälkitunnisteessa käytettäviä ominaisuuksia ja niiden keskimääräinen elinikä %:lla käyttäjistä. (Vastel et al., 2018)

kustaessa aikavyöhykkeeltä toiselle tai ottaessa käyttöön ulkoisen näytön, jolloin todettava resoluutio muuttuu. Käyttäjän itse aiheuttamat muutokset (user-triggered evolutions) liittyvät asetuksiin, esimerkiksi käyttäjän salliessa evästeet tai vaihtaessa käyttöjärjestelmän kielen.

Näistä itsestään muuttuvat ominaisuudet olivat useimmiten eliniältään lyhyimpiä, sillä selaimet päivittyvät usein. Ominaisuudet voivat kuitenkin olla lyhyestäkin elinajasta huolimatta käyttökelpoisia sormenjälkitunnistamisessa, sillä esimerkiksi selaimen version vaihtuminen uudempaan on helposti selitettävissä. Toisten ominaisuuksien, kuten näytön resoluution, vaihtuminen on vaikeampi selittää. Sormenjälkitunnisteiden tai niissä käytettävien ominaisuuksien yhdistämisen helppoudesta aiempaan versioon puhutaan niiden linkitettävyydestä (linkability). (Vastel et al., 2018)

Vastel et al. rakensivat algoritmin, jolla sormenjälkitunnisteet voidaan yhdistää aiempaan versioon ottamalla huomioon ominaisuuksien muuttuminen aikaa myöten. Algoritmin säännöt sallivat helposti ymmärrettävät muutokset esimerkiksi selaimen versiosta uudempaan, mutta rajoittavat vaikeammin selitettäviä muutoksia sallimalla vain tietyn määrän vaikeasti linkitettäviä muutoksia yhdellä kerralla. Jos muutoksia on liikaa, eli sormenjälkitunnistetta ei voida linkittää aiempaan, nähdään uusi sormenjälkitunniste uutena käyttäjänä.

Sormenjälkitunnisteen elinaikaa on mahdollista pidentää rajoittamalla ominaisuuksia pitkäaikaisiin tai käyttämällä sormenjälkitunnisteen linkitettävyyttä parantavaa algoritmia. Elinajan pidentäminen vähentää sormenjälkitunnistusmenetelmän yksilöllistävyyttä, mutta voi toisaalta mahdollistaa pitkäaikaisenkin käyttäjän seurannan.

6.13 Sormenjälkitunnisteen yksilöllistävyys

Vaikka tarkoitus on usein viime kädessä yksilöidä käyttäjä, voi yhtä sormenjälkitunnistetta vastata useampi käyttäjä. Sormenjälkitunnisteet toimivat mitattavien ominaisuuksien mahdollisten arvojen monipuolisuudesta riippuen jakamalla käyttäjiä yhä pienempiin anonymiteettijoukkoihin, jotka jakavat yhden sormenjälkitunnisteen (Eckersley, 2010). Siksi sormenjälkitunnisteen pitää olla mahdollisimman tehokas, jos sitä käytetään tilallisten menetelmien korvikkeena. Sormenjälkitunniste on yksilöllinen, kun sen anonymiteettijoukon koko on yksi.

Sormenjälkitunnistuksen tehokkuutta on mitattu useissa tutkimuksissa, joista laajimmat lienevät ‘Panopticlick’ (Eckersley, 2010) ja ‘AmIUnique’ (Laperdrix et al., 2016), sekä Gómez-Boix et al. (2018). ‘Panopticlick’ ja ‘AmIUnique’ -tut-

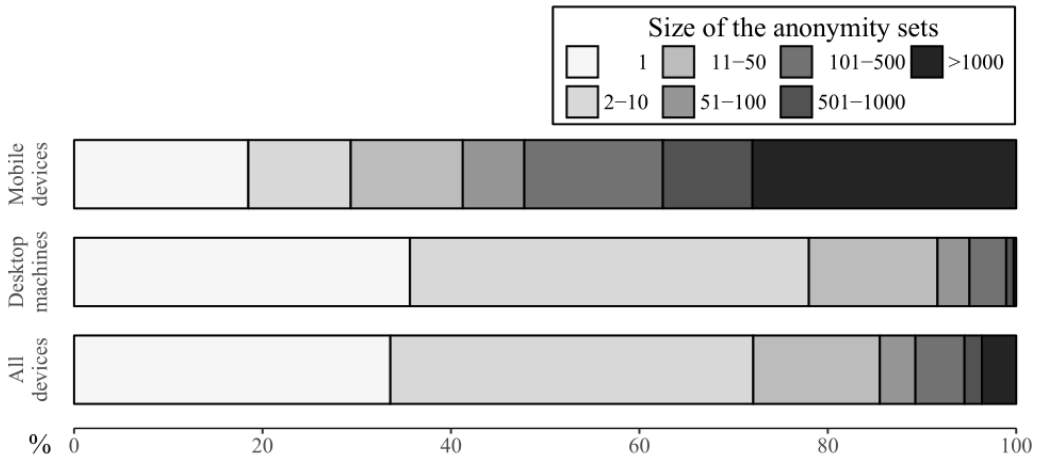
kimuksissa tutkijat loivat sormenjälkitunnisteiden keräämiselle ja esittelemiselle uuden verkkosivun, jonne he kutsuivat käyttäjiä luovuttamaan oman selaimen sormenjälkitunnisteensa. Gómez-Boix et al. puolestaan keräsivät käyttäjien sormenjälkitunnisteita suositulla verkkosivustolla käyttäjien asiasta tietämättä.

'Panopticlick'-tutkimukseen kerättiin yhteensä 470 161 sormenjälkitunnistetta, joista yksilöllisiä oli 83,6%. Tutkimuksen sormenjälkitunnisteessa käytettiin `User-Agent` -otsaketta, näytön resoluutiota, aikavyöhykettä, käytössä olevia fontteja, selaimen liitännäisiä, sekä sitä, onko evästeiden käyttö asetettu pois päältä. Yksilöivimpiä ominaisuuksia olivat `User-Agent`, fontit sekä liitännäiset. Kuvassa 6.2 on tutkimuksissa ominaisuuksille lasketut entropiat. Niiden käyttäjien, joilla Java- tai Flash-liitännäinen oli käytössä, sormenjälkitunnisteista jopa 94,2% oli yksilöllisiä. Liitännäisen avulla muun muassa fonttien listaus oli yksilöivämpi, kuin JavaScriptillä saman tiedon hakeminen, johtuen listan tarkkuudesta ja pysyvästä järjestyksestä liitännäistä käyttäessä. Toisaalta 'Panopticlick'-tutkimus tehtiin vuonna 2010, ja sittemmin liitännäisten käyttö ja selainten tuki liitännäisille on vähentynyt, vähentäen niiden lisäämää yksilöllisyyttä (Gómez-Boix et al., 2018).

Laperdrix et al. keräsivät 'AmIUnique' -tutkimukseen vuonna 2016 yhteensä 118 934 sormenjälkitunnistetta, joista he havaitsivat 89,4% olevan yksilöllisiä. Tutkijat huomioivat sormenjälkitunnistusmenetelmässään 'Panopticlick'-tutkimusta isomman joukon ominaisuuksia, lisäten muun muassa Canvas-sormenjälkitunnisteen, selaimen kielen, näytönohjaimen tiedot, Do Not Track -asetuksen, mainosten estämisen käytön ja monia muita selaimen ominaisuuksia. Liitännäisiä itseään ei enää käytetty tutkimuksessa tarkemman tiedon, muun muassa fonttien, hakemiseen. Liitännäisten tuki selaimista on myös ollut poistumassa jo tutkimusta tehtäessä, mutta muiden uusien ominaisuuksien lisääminen sormenjälkitunnisteisiin piti niiden yksilöllisyyden edelleen hyvin korkealla. Laperdrix et al. tekivät myös ensimmäisen laajan tutkimuksen mobiililaitteiden sormenjälkitunnisteista, ja havaitsivat aineistossaan 81% niistä olevan yksilöllisiä.

Gómez-Boix et al. (2018) kritisoi 'Panopticlick' ja 'AmIUnique' -tutkimuksia niiden kohderyhmästä johtuen. Tekemällä verkkosivuston sormenjälkitunnisteiden keräämiseen tutkimukseen osallistuneet käyttäjät olivat luultavasti kiinnostuneita yksityisyydestä ja teknologiasta. Sormenjälkitunnisteiden esittelyyn keskittynyt sivusto myös kannusti käyttäjiä selaimen asetuksien ja kokoonpanon vaihtelemiseen nähdäkseen sen vaikutuksen laskettuun sormenjälkitunnisteeseen.

Kahdesta aikaisemmasta tutkimuksesta poiketen Gómez-Boix et al. käyttivät tutkimusaineiston keräämiseen yhtä Ranskan suosituimmista verkkopalveluista.



Kuva 6.7: Sormenjälkitunnistamisen tehokkuutta mittaavassa tutkimuksessa kerättyjen sormenjälkitunnisteiden anonymiteettijoukkojen kokojen osuudet mobiililaitteissa, tietokoneissa ja kaikissa laitteissa. Mobiililaitteiden anonymiteettijoukot olivat paljon isompia, esimerkiksi yli 50 laitteen anonymiteettijoukkoja oli mobiililaitteista noin 59% sormenjälkitunnisteista, kun tietokoneista yhtä isoja oli vain noin 8% laitteista. (Gómez-Boix et al., 2018)

Sormenjälkitunnisteita he keräsivät yhteensä 2 067 942. Yksilöllisiä sormenjälkitunnisteista oli aikaisemmista tutkimuksista poiketen 33,6%. Tietokoneita ja mobiililaitteita erikseen tarkastellessa vastaavat luvut olivat 35,7% ja 18,5%. Sormenjälkitunnisteiden anonymiteettijoukkojen koot ovat kuvattuna kuvassa 6.7, josta myös huomaa mobiililaitteiden sormenjälkitunnisteiden jakautuvan selvästi suurempiin anonymiteettijoukkoihin. Suuremmat anonymiteettijoukot johtuvat muun muassa selaimen liitännäisten puuttumisesta mobiililaitteilla.

Tuloksien suuri ero aikaisempiin verrattuna arveltiin liittyvän aineistoon kerättyjen sormenjälkitunnisteiden suurempaan määrään, keskimääräistä käyttäjää vastaavampaan kohderyhmään, liitännäisten käytön jatkuvaan vähentymiseen ja käyttäjien maantieteellisellä sijoittumisella Ranskaan. Tutkimusaineisto kerättiin yhdellä Ranskan suosituimmista verkkopalveluista, joten suurimmalla osalla käyttäjistä olivat selaimen kieli ja aikavyöhyke sen mukaan asetettu. Lisäksi Gómez-Boix et al. mainitsevat esimerkiksi maantieteellisen yleisön vaikuttavan myös laitteiden ja selainten käyttöjakaumiin.

Gómez-Boix et al. pitivät kuitenkin tuloksia yksityisyyden puolesta lupaavina aikaisempiin tutkimustuloksiin verrattuna. He kävivät myös joitakin skenaarioita, joilla selaimien sormenjälkitunnisteet saisi yleisemmiksi. Esimerkiksi liitännäisten

poistuminen kokonaan käytöstä vähentäisi tietokoneiden sormenjälkitunnisteista yksilöllisiksi 16,5%. Vähemmän realistisessa skenaariossa JavaScriptin poistuminen käytöstä pudottaisi sormenjälkitunnisteista yksilöllisiksi vain 4,3%, sillä sormenjälkitunnistamiseen olisi enää käytettävissä pelkästään käyttäjän passiivinen, palvelimella kerättävä, sormenjälkitunniste.

Tutkimukset eivät kuitenkaan mittaa kaikkia mahdollisia sormenjälkitunnisteissa käytettäviä ominaisuuksia. Esimerkiksi `IP`-osoite, Web Audio ja Cao et al.:n (2017) tutkima WebGL-sormenjälkitunniste eivät ole näissä tutkimuksissa käytössä, joten niitä käyttämällä sormenjälkitunnistuksen yksilöllistävyys luultavasti parantuisi.

6.14 Zombievästeet ja selaimen sormenjälkitunniste.

Tilattomia menetelmiä voidaan käyttää yhtenä zombievästeen komponenttina. Zombievästeen toiminta perustuu useamman tallennusmekanismin hyödyntämiseen, jotta sen poistaminen olisi mahdollisimman hankalaa. Zombievästeen tallenteet pyritään asettamaan uudelleen, jos joitakin niistä on poistettu. (Acar et al., 2014)

Listauksessa 6.1 on esimerkki käyttäjän tunnisteen asettamisesta uudelleen evästeisiin, kun hänen sormenjälkitunnisteensa on vastaanotettu ja aikaisemmin otettu talteen. Tilattomia menetelmiä voi siis käyttää myös zombievästeiden yhtenä komponenttina ja ne mahdollistavat tunnisteen palauttamisen, vaikka kaikki tallenteet olisivatkin poistettu käyttäjän koneelta. Sormenjälkitunnisteen ei tarvitse myöskään välttämättä olla pitkäaikainen, jos sitä käytetään käyttäjän tunnisteen palauttamiseen pian sen jälkeen kun käyttäjä on tyhjentänyt kaikki selaimestaan evästeet, välimuistin ja muut tallenteet.

6.15 Somenjälkitunnistuksen muita käyttöjä

Selaimen sormenjälkitunnistusta käytetään yleisesti myös käyttäjän todentamisessa salasanakirjautumisen tukena. Kirjautumisen yhteydessä selaimen sormenjälkitunniste otetaan talteen, jolloin myöhemmällä istunnolla sen hetkistä selainta voidaan verrata aiemmin talletettuun sormenjälkitunnisteeseen. Kaksivaiheisessa tunnistuksessa käyttäjää, jonka sormenjälkitunnistetta ei voida tunnistaa samaksi kuin aikaisemmilla kerroilla käytetyn selaimen sormenjälkitunniste, pyydetään käyttämään ylimääräistä tunnistusmenetelmää henkilöllisyyden varmistamiseksi. Vaadittu lisätunnistusmenetelmä voi olla esimerkiksi tekstiviestiin tai sähköpos-

```
/**
 * AJAX endpoint for receiving computed browser fingerprint.
 * @param req HTTP request object
 * @param res HTTP response object
 */
function cookieRefreshingReceiveFingerprint(req, res) {
    // Fingerprint received from user in AJAX request body.
    const fingerprint = req.body;
    if (!fingerprint) {
        res.status(200).end();
        return;
    }
    const found_nickname = FINGERPRINTS[fingerprint];
    // If fingerprint found from the database use it to refresh cookie.
    if (found_nickname) {
        console.log(`Fingerprint found for ${found_nickname}.`);
        res.cookie("nickname", found_nickname, {
            path: "/cookie-refreshing",
            expires: new Date(Date.now() + 1000 * 60 * 60 * 24)
        });
        res.send(found_nickname);
        return;
    }
    const nickname = req.cookies["nickname"];
    // If cookies contain identifier map it to received fingerprint.
    if (nickname) {
        console.log(`Setting fingerprint ${fingerprint} for ${nickname}.`);
        FINGERPRINTS[fingerprint] = nickname;
    }
    res.status(200).end();
}
```

Listaus 6.1: Yksinkertainen sormenjälkitunnisteen hyödyntäminen evästeen päivityksessä Express-verkkosovelluskehyksellä. Kun sormenjälkitunniste otetaan vastaan, sitä verrataan aiemmin lisättyihin sormenjälkitunnisteisiin. Jos sormenjälkitunniste löytyy aiemmin lisätyistä, voidaan eväste päivittää sen mukaisesti. Muussa tapauksessa evästeen löytyessä voidaan sormenjälkitunniste lisätä tunnettujen joukkoon.

tiin tilattava varmistuskoodi. Onnistuneen varmistuksen jälkeen sormenjälkitunniste voidaan ottaa talteen, ja myöhemmillä kirjautumiskerroilla sormenjälkitunnisteiden täsmätessä erillisen tunnistusmenetelmän käyttö ohittaa. (Alaca ja van Oorschot, 2016)

Sormenjälkitunnistuksen avulla käyttäjien todentamista voi suojata sekä sisäänkirjautumisvaiheessa kuin istunnossa. Alacan ja van Oorschotin (2016) mukaan sillä voidaan ehkäistä kirjautumisvaiheessa satunnaisiin tai yksittäisiin tunnuksiin kohdistettua väsytyshyökkäystä, jossa salasanoja pyritään arvaamaan, sekä yksittäisiin tunnistuksiin kohdistettuja hyökkäyksiä, joissa käyttäjän salasana on jo aikaisemmin tiedossa. Sormenjälkitunnistusta hyödyntäessä kirjautuminen vaatii enemmän työtä hyökkääjältä, sillä tämän pitää selvittää ja toistaa salasanan lisäksi myös käyttäjän sormenjälkitunniste. Myös istuntokaappauksen uhkaa voidaan sormenjälkitunnisteen avulla ehkäistä vertaamalla käyttäjän sormenjälkitunnistetta aikaisemmin talletettuun kesken istunnon.

7 Seurannan torjunta

Käyttäjän suojautuminen verkossa tapahtuvalta seurannalta voi vaikeuttaa verkon käyttöä ja rajoittaa toiminnallisuuksia suojautumisen tehokkuudesta riippuen. Tilallisissa seurantamenetelmissä käytettäviä tallenteita, kuten evästeitä, käytetään muun muassa verkkopalveluiden asetuksien säilyttämiseen ja asiakastilille kirjautumiseen. Myös kolmannen osapuolen evästeet voivat olla käytössä esimerkiksi kirjautumisessa tai tiedon siirrossa verkkopalveluiden välillä. Jos tallenteet tyhjennetään, tulee käyttäjän kirjautua tai valita asetukset uudelleen palvelua käyttäessään, hidastaen verkon käyttöä. Tallenteet kokonaan poistettuna käytöstä monet toiminnallisuudet eivät tule toimimaan ollenkaan.

Tilattomilta menetelmiltä suojautuessa vaikutetaan suoraan selaimen toimintoihin. Esimerkiksi käytettäviä fontteja voidaan rajoittaa pienempään ja yleisempään joukkoon fontteja, jotta fonttien joukko yhtenäisempi muiden käyttäjien kanssa. Toisaalta tämä tarkoittaa, että käytöstä rajatut fontit eivät ole enää verkkopalveluiden käytettävissä. Canvasin 'ToDataURL'-metodin ulostuloon voidaan vaikuttaa pienillä käyttäjälle huomaamattomilla muutoksilla, jotta se olisi satunnainen joka käyntikerralla, tai vaatia metodin käyttöön käyttäjän hyväksynnän. Jotkut yksityisyyden uhkana nähdyt toiminnallisuudet saatetaan ottaa selaimesta kokonaan pois käytöstä, kuten Tor Browser -selain on tehnyt esimerkiksi äänisormenjälkitunnistamisen mahdollistavan Web Audio rajapinnan kanssa. Lisäksi sormenjälkitunnistamisen estäminen voi vaikuttaa muun muassa edellisessä luvussa esiteltyyn tapaan toistuvien käyntikertojen todentamiseksi. Tällöin kaksivaiheista tunnistusta käytettäessä käyttäjä saattaa joutua todentamaan itsensä jokaisella käyntikerralla ylimääräistä tunnistusmenetelmää käyttäen.

Käyttäjä voi myös estää käyttäjää jäljittävien resurssien latautumisen. Verkon mainonta sisältää usein käyttäjää jäljittäviä toiminnallisuuksia, joten mainoksia estävät estolistat ovat tulleet käyttöön myös käyttäjän seurannan estämisessä. Resurssien lataamisen estäminen voi vähentää käyttäjän altistumista seurannalle. Toisaalta estolistojen käyttö voi olla verkkopalvelun huomattavissa.

Tor Browser -selainta on kehitetty torjumaan käyttäjän seurantamenetelmiä, ja se listaa yksityisyyden suojan toteuttamiseen kolme tavoitetta: Käyttäjä ei saa olla linkitettävissä sivustojen välillä tallenteiden, esimerkiksi kolmannen osapuolen evästeiden tai välimuistievästeiden avulla. Käyttäjää ei myöskään saa olla mahdollista seurata sivustojen välillä sormenjälkitunnisteen avulla. Koska sormenjälkitunniste pysyy samana vierailtavasta sivustosta riippumatta, pyrkii Tor Browser minimoimaan sormenjälkitunnisteen yksilöivyyttä seurannan ehkäisemi-

seksi. Viimeinen tavoite liittyy käyttäjän pitkäaikaiseen seurantaan. Selaimen tulee tarjota helppo tapa käyttäjälle poistaa tunnistee, jotta hän voi rajoittaa pitkäaikaista seurantaa. Tallenteiden tyhjennys tulisi myös tapahtua oletuksena selaimen suljettua.

Käyn läpi eri tapoja, joilla käyttäjä voi pyrkiä estämään verkossa tapahtuvaa seurantaa. Alussa esiteltävät Do Not Track -otsake ja opt-out evästeet toimivat lähettämällä verkossa seurantaa harjoittavalle taholle pyynnön seurannan lopettamiseksi. Muut keinot toimivat selaimen toiminnallisuuksia muuttamalla tallenteilla ja sormenjälkitunnistuksella tapahtuvan seurannan estämiseksi.

7.1 Do Not Track

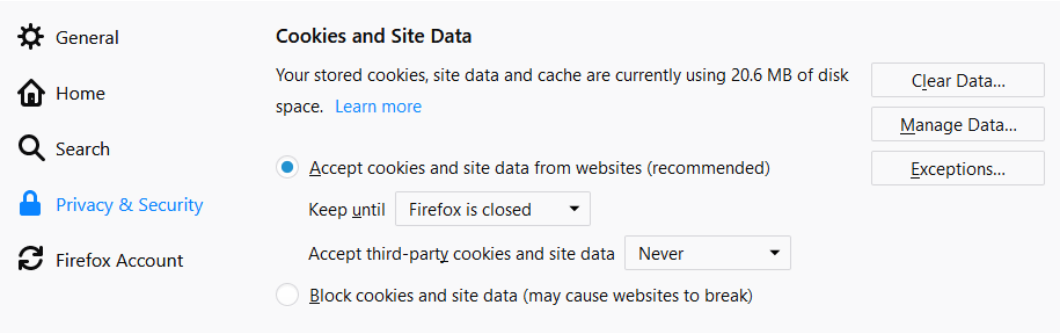
Kuten luvussa 6.4.4 mainittiin, voi käyttäjä pyytää verkkopalvelua olemaan seuraamatta käyttäjää asettamalla Do Not Track -asetuksen selaimesta päälle, joka lähettää pyynnön verkkopalvelulle ylimääräisenä `HTTP`-otsakkeena.

Koska pyyntö ei takaa verkkopalvelulta mitään toimia sen noudattamiseen, Do Not Track -asetuksen käytön vaikutusta käyttäjän seuraamiseen on hankala arvioida. Muun muassa Acar et al. (2013) arvioivat sormenjälkitunnistusta käyttävien palveluiden sivuuttavan pyynnön. Myös Microsoft ilmoittaa tietosuojatekstissään, että se ei reagoi otsakkeeseen millään tavalla, sillä yritys ei näe vielä yleistä käsitystä siitä, mitä Do Not Track -pyynnön noudattaminen tarkoittaisi (Microsoft, 2016).

7.2 Mainostajaverkkojen opt-out evästeet

Monet suuret mainostajaverkot ja mainostajien ryhmittymät tarjoavat mahdollisuuden käyttäjälle kieltää kohdistetun mainonnan, eli OBA:n (online behavioral advertising), vastaanottamisen. Käyttäytymistietoa, kuten selaushistoriaa, kerätään käyttäjästä usein kohdistettua mainontaa varten, joten kiellon antaminen mainostajalle voi olla yksi tapa estää verkon seurantaa.

Kohdennetun mainonnan kieltäminen (opt-out) tapahtuu usein erillisissä verkkopalveluissa, joissa voidaan valita kyseisen mainostajan, tai mainostajien ryhmittymän yrityksistä ne, joiden kohdistetulta mainonnalta halutaan välttyä. Esto toimii evästepohjaisesti, eli käyttäjän selaimeen tallennetaan eväste kaikkiin palveluihin, joihin kielto asetetaan. Käyttäjän vieraillessa palvelussa, tai ladatessa sisältöä palvelusta, johon kielto on annettu, voi palvelu huomatessaan evästeen olemassaolon olla ottamatta käyttäjän tietoa talteen. (Leon et al., 2012)



Kuva 7.1: Firefoxin yksityisyysasetukset, joilla voi muun muassa estää kaikkien evästeiden käytön tai kolmannen osapuolen evästeiden asettamisen, tai rajoittaa evästeiden elinaika selaimen sulkemiseen.

Kohdennetun mainonnan kieltojen käyttö on kuitenkin työlästä, sillä mainostajaverkkoja on paljon, ja käyttäjän tehtäväksi jää hakea kieltopyyntö mahdollisimman monelle mainostajaverkolle. Evästeisiin perustuva kieltopyyntö tulee myös hakea aina uudestaan, jos selaimen evästeet tyhjennetään tai vanhentuvat. (Mayer ja Mitchell, 2012)

Do Not Trackin tapaan kiellon noudattaminen ja soveltaminen jää mainostajan tehtäväksi. Monissa tapauksissa kohdistetun mainonnan ja käyttäjän seuraaminen nähdään erillisinä toimina. Mainostajaverkot eivät myöskään ole ainoa käyttäjää verkossa seuraava taho, joten kieltojen käyttö voi jäädä puutteelliseksi keinoksi estää seurantaa. (Mayer ja Mitchell, 2012)

7.3 Tallenteiden estäminen

Selaimet antavat mahdollisuuden estää evästeiden ja muiden tallenteiden käytön kokonaan tai osittain. Kun evästeiden käyttö on estetty, ei selain tallenna palvelimelta vastaanotettuja evästeitä. Firefoxin ‘Block cookies and site data’ valinta estää myös Web Storagen ja muiden asiakaspuolen tallennusmekanismien käytön (kuva 7.1). Lisäämällä tarvitsemansa palvelut erilliselle poikkeuslistalle, voi käyttäjä kuitenkin antaa verkkopalvelulle oikeudet tallenteiden käyttöön. Tallenteet estettynä ja poikkeuslistaa hyödyntäen käyttäjä voi minimoida mahdollisten seurantatunnisteiden tallentumisen vain niihin palveluihin, jotka hän on sallinut. (Mozilla, 2012b)

Evästeiden kokonaan estäminen voi tehdä verkon käyttämisen vaivalloiseksi, sillä se voi estää esimerkiksi verkkopalveluun kirjautumisen tai verkkokaupan

ostoskorin käytön. Iso osa verkon seurannasta toimii kuitenkin kolmannen osapuolen evästeillä, joten useimmissa selaimissa on mahdollisuus estää näiden toimintaa erikseen. Kun Firefox-selaimessa estää kolmannen osapuolen evästeiden asettamisen, ei se enää tallenna kolmannen osapuolen resurssien mukana lähetettyjä evästeitä. (Mozilla, 2012a)

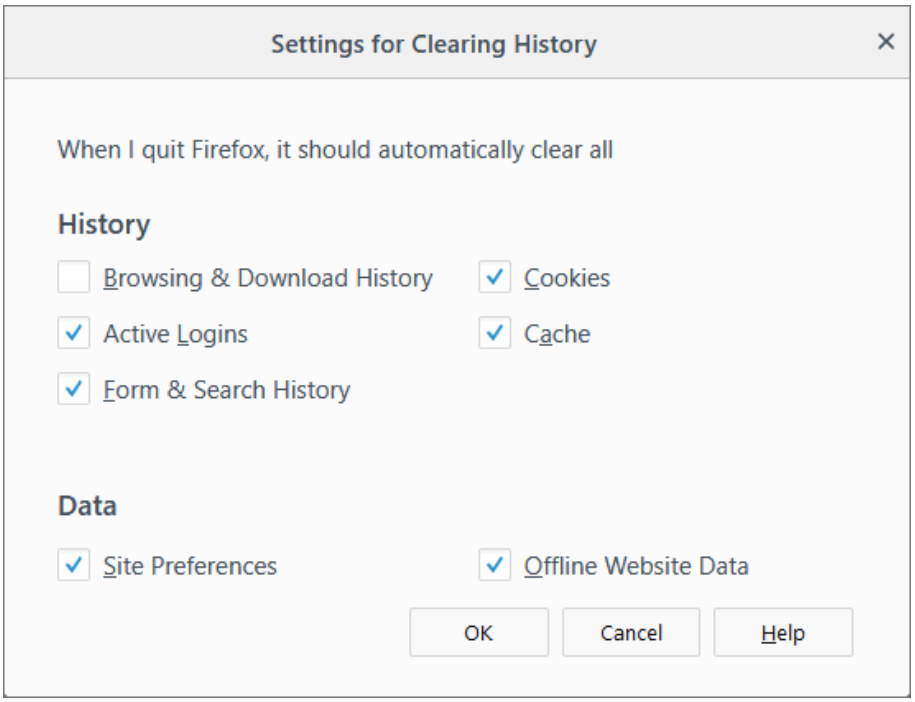
Vaikka kolmannen osapuolen evästeiden asettaminen on estetty, se ei kuitenkaan vaikuta tapauksiin, jossa kolmannen osapuolen evästeet ovat jo aikaisemmin asetettu. Aikaisemmin asetetut kolmannen osapuolen evästeet välittyvät kolmannelle osapuolelle normaalisti. Verkkopalvelut voivat käyttää esimerkiksi uudelleenohjausta kolmannen osapuolen resurssiin, jolloin kolmannella osapuolella on mahdollisuus asettaa eväste ensimmäisen osapuolen sijalta. Muita tapoja kiertää kolmannen osapuolen evästeiden esto on ponnahdusikkunalla. Kolmannen osapuolen verkkopalvelusta tarjottu sivusto on ponnahdusikkunassa ensimmäisen osapuolen asemassa, jolloin evästeet voidaan asettaa vaikka kolmannen osapuolen evästeiden asettaminen olisikin estetty. (Mayer ja Mitchell, 2012)

Monet käyttäjiä verkossa seuraavat palvelut ovat asemassa, jossa käyttäjillä voi olla evästeet asetettuna aikaisemmalta vierailulta. Esimerkki tällaisesta palvelusta on Facebook. Yhteisöpalvelun käyttäjien vieraillessa muualla verkossa, jossa Facebookin tarjoamaa sisältöä, muun muassa tykkäysnappuloita, on käytössä, välittyvät evästeet normaaliin tapaan Facebookille. Tällaisessa tapauksessa kolmannen osapuolen evästeiden asettamisen esto ei auta seurannalta suojautumiseen. (Mayer ja Mitchell, 2012)

7.4 Tallenteiden ajoittainen poistaminen

Käyttäjä voi vaikuttaa seurannan kestoon tyhjentämällä säännöllisesti selaimen evästeitä ja muita tallenteita. Selaimen asetuksilla ja lisäosilla voi ajoittaa tallenteiden poistamista tiettyihin ajankohtiin, esimerkiksi selaimen sulkemiseen tai sivustolta poistumiseen. Firefox-selaimen asetuksilla tallenteet voi asettaa poistettavaksi ennen kuin selain suljetaan (Mozilla, 2014d). Asetusikkuna on nähtävillä kuvassa 7.2. Tyhjentämällä kerralla evästeet, välimuistin ja asiakaspuolen tallennusmekanismit käyttäjä pääsee varmemmin eroon mahdollisista zombievästeen tapaisista käyttäjätunnisteista. Toisaalta tällöin käyttäjän tulee tehdä tarvitsemansa verkkopalveluasetukset ja -kirjautumiset aina avatessaan selaimen uudestaan.

Selaimien lisäosarajapinnoilla voi toteuttaa edelleen vaihtoehtoisia strategioita tallenteiden poistamiseksi. Cookie AutoDelete -lisäosa poistaa sivuston tallen-



Kuva 7.2: Firefoxin asetuksilla on mahdollista tyhjentää selaushistoria, tallenteet ja sivustokohtaiset selainasetukset selainta sulkiessa.

teet heti kun käyttäjä on sulkenut auki kyseisen sivuston välilehden (Do, 2017). Käyttäjän ei siis tarvitse tyhjentää evästeitä manuaalisesti tai käynnistää selainta uudelleen. Lisäksi tallenteet poistetaan sivustokohtaisesti eli yhden sivuston tallenteiden tyhjentäminen ei vaikuta muilla välilehdillä auki oleviin sivustoihin.

7.5 Tallenteiden eristäminen

Yksi suurimpia yksityisyyden riskejä on sivustojen välinen seuraaminen, sillä se voi paljastaa käyttäjän selaushistoriaa. Sivustojen välinen seuranta toimii usein evästeitä käyttämällä, kun selain hakee resurssia kolmannelta osapuolelta, jolle eväste on jo aikaisemmin asetettu. Resurssia hakiessa välittyy tietoa käyttäjällä auki olevasta sivustosta. Sama eväste on myös käytössä muilla sivustoilla, jotka käyttävät samaa kolmannen osapuolen resurssia. (Luku 5.3)

Toisaalta käyttäjä voi myös haluta käyttää jotakin verkkopalvelua useammalla ‘identiteetillä’, eli erillisillä evästeillä, jotka voivat sisältää asetuksia, kirjautumisen tai muuta yksilöivää tietoa. Google Chrome sisältää profiilit, joilla käyttäjä voi esimerkiksi erotella henkilökohtaisen ja työhön liittyvän verkon käytön toisistaan (Google, 2013). Profiilit ovat eristettynä toisistaan, eli ne sisältävät omat

evästeet, välimuistin ja muut tallenteet. Näin käyttäjä voi käyttää samaa verkkopalvelua useammalla eri identiteetillä ilman, että verkkopalvelu voi yhdistää identiteettejä toisiinsa ainakaan tallenteita käyttämällä. Lisäksi käyttäjä voi vähentää sivustojen välisen seurannan kattavuutta käyttämällä eri profiileja verkon eri palveluiden käyttämiseen.

Mozillan 'Facebook Container' -lisäosalla voi eristää Facebook-yhteisöpalvelun evästeet muusta verkon selauksesta (Mozilla, 2018). Huomattava osa verkon sivustoista käyttää Facebookin tykkäysnappuloita, kommenttilaatikoita ja muita palveluita (BuiltWith, 2018). Facebook pääsee siksi käsiksi miljoonien Facebookin käyttäjien kattavaan selaushistoriaan, sillä samat evästeet ovat käytössä itse Facebook.com-palvelussa, kuin muualla verkossa, jossa Facebookin kolmannen osapuolen palvelut ovat käytössä. Lisäosan avulla Facebookin evästeet itse palvelua käyttäessä eristetään käyttäjän selatessa muita sivustoja verkossa. Näin käyttäjän selaushistoria ei suoraan välity Facebookille.

Joissain selaimissa, kuten Firefoxissa, on myös mahdollista eristää selaimen tallenteet kokonaan sen mukaan, millä ensimmäisen osapuolen sivustolla vieraillessa ne on asetettu. Asetus estää tallenteiden, muun muassa evästeiden ja välimuistin, jakamisen kolmannen osapuolen resursseja hakiessa eri verkkopalveluita käyttäessä, estäen näiden käytön sivustojen väliseen seurantaan (Mozilla, 2017b). Siinä missä tavallisesti evästeet tallennetaan selaimeen sen asettaneen domain-osoitteen mukaan, otetaan asetusta käyttäessä huomioon myös käyttäjän vieraileva sivusto, eli ensimmäisen osapuolen verkkopalvelun domain-osoite (Perry et al., 2013). Tallenteet toisistaan eristävä toiminto on kuvattu evästeiden osalta kuvassa 7.3. Firefoxissa asetuksen nimi on 'firstPartyIsolation' ja se on toteutettu 'Tor Uplift' -projektissa (Mozilla, 2017b; Steele, 2016). 'Tor Uplift' -projektissa Tor Browser -selaimen yksityisyyteen liittyviä toiminnallisuuksia tuodaan Firefox-selaimeen käytettäväksi. Kerron projektista lisää myöhemmin kappaleessa 7.12. Tallenteiden eristäminen kuitenkin vaikeuttaa jonkin verran verkkopalveluiden käyttöä, sillä monet palvelut käyttävät esimerkiksi kolmannen osapuolen evästeitä kirjautumiseen verkkopalveluiden välillä (Mozilla, 2017b).

7.6 Sormenjälkitunnisteen satunnaistaminen ja yhdenmukaistaminen

Sormenjälkitunnistukselta suojautuminen voidaan jakaa kahteen eri tapaan: satunnaistamiseen ja yhdenmukaistamiseen (Perry et al., 2013). Molemmat pyrkivät vaikuttamaan sormenjälkitunnisteen linkitettävyyteen, mutta tekevät sen eri tavoilla. Sormenjälkitunnisteen satunnaistamisella pyritään antamaan verkkopal-

Resurssin domain-osoitetta avaimena käyttäen:

Domain	Vierailtu domain	Eväste
facebook.com	facebook.com	eväste-1
facebook.com	hs.fi	eväste-1
facebook.com	il.fi	eväste-1

Resurssin domain-osoitetta, ja käyttäjän vierailemaa domain-osoitetta, avaimena käyttäen:

Domain	Vierailtu domain	Eväste
facebook.com	facebook.com	eväste-1
facebook.com	hs.fi	eväste-2
facebook.com	il.fi	eväste-3

Kuva 7.3: Kuvassa käyttäjä lataa sisältöä Facebookista yhteisöpalvelun sivustolla ja muualla verkossa. Tavallisesti eväste käyttää avaimena ladattavan resurssin domain-osoitetta, jolloin sama eväste on käytössä riippumatta siitä, missä verkkopalvelussa käyttäjä vierailee. Firefoxin 'firstPartyIsolation'-asetusta tai Tor Browser -selainta käytettäessä evästeiden avain on yhdiste resurssin domain-osoitteesta ja käyttäjän vieraileman sivuston domain-osoitteesta, jolloin evästettä ei jaeta, estäen sivustojen välistä seurantaa.

velun käyntikerralla tai eri verkkopalveluissa käydessä erilainen sormenjälkitunniste. Sormenjälkitunnisteen pitää olla tarpeeksi erilainen, jotta sitä ei voida yhdistää aikaisempiin tai muissa verkkopalveluissa annettuihin sormenjälkitunnisteisiin. Rikkomalla linkitettävyyden sormenjälkitunnisteiden välillä käyttäjä saa aikaan sen, että verkkopalvelut eivät voi tunnistaa käyttäjää samaksi (Nikiforakis et al., 2015).

Sormenjälkitunnisteen satunnaistamisella on kuitenkin rajoitteita, koska monet sormenjälkitunnisteeseen kerättävät ominaisuudet ovat haettavissa useampaa reittiä. Esimerkiksi käyttäjän käyttöjärjestelmä on selvitettävissä `HTTP`-kutsun `User-Agent`-otsakkeesta, asiakaspuolelta muun muassa `navigator`-rajapinnasta sekä `TCP`-liikenteestä. Epäjohdonmukaisesti ilmoitettuja tietoja voidaan käyttää lisäinformaationa sormenjälkitunnisteessa (Nikiforakis et al., 2013).

Satunnaistaessa sormenjälkitunnistetta pitää myös huomioida, että sormenjälkitunnisteeseen kerättävien ominaisuuksien tulee olla mahdollisia (Nikiforakis et al., 2015). Esimerkiksi 'Ubuntu 16.04'-käyttöjärjestelmän ja 'Safari'-selaimen yhteiskäyttö on mahdoton yhdistelmä, sillä Ubuntulle ei ole tarjolla kyseistä selainta. Mahdottoman yhdistelmän käyttö tekee sormenjälkitunnisteesta hyvinkin yksilöllisen, sillä harvalla käyttäjällä tulee olemaan sama yhdistelmä käytössä.

Sormenjälkitunnistetta yhdenmukaistaessa pyritään suurentamaan käyttäjän kanssa samanlaisen sormenjälkitunnisteen omaavien käyttäjien määrää eli sormenjälkitunnisteen anonymiteettijoukon kokoa. Kun selain pyrkii yhdenmukaistamaan sormenjälkitunnisteet käyttäjien välillä, pienentyy sen yksilöivyys ja käyttäjän erottaminen toisesta käyttäjästä vaikeutuu. Suurin osa tässä tutkielmassa esitetyistä vastakeinoista sormenjälkitunnistuksen tehokkuuden vähentämiseksi liittyy sormenjälkitunnisteen yhdenmukaistamiseen. Toisaalta yksityisyys on hyvin sidottu saman sormenjälkitunnisteen omaavien käyttäjien määrään, koska selain ei satunnaistamalla sormenjälkitunnistetta pyri rikkomaan linkitettävyyttä aikaisempaan vierailuun tai muuhun verkon käyttöön. (Perry et al., 2013)

Perry et al. (2013) päätyivät Tor Browser -selainta kehittäessä sormenjälkitunnisteen yhdenmukaistamiseen. Syiksi he mainitsevat muun muassa satunnaistetun sormenjälkitunnisteen tehokkuuden mittauksen vaikeuden, monien ominaisuuden satunnaistamisen vaikeuden, toiminnallisuuksien vaihtelevuuden satunnaistumisen mukaan, ja mahdolliset tietoturva-aukot, joita satunnaistamisen toteuttaminen voi luoda. Tor Browser pyrkii yhdenmukaistamaan monia sormenjälkitunnisteissa käytettäviä ominaisuuksia kuten fontit, näytön resoluution ja näppäimistön asettelun. Monet yksilöivät ominaisuudet, kuten liitännäiset ja Canvas ovat pois käytöstä ainakin oletuksena, jolloin nekin ovat mahdollisimman yhtenäisiä

muiden Tor Browser -selaimen käyttäjien kanssa. Tor Browser -selaimesta kerron lisää kappaleessa 7.12.

7.7 Käyttöjärjestelmän tai selaimen nimen ja version vaihtaminen

Selaimen lisäosilla on mahdollista korvata selaimen oletuksena lähettämä `User-Agent`-otsake. Otsake sisältää käyttöjärjestelmän ja selaimen nimet ja versiot. Selaimen lisäosalla tai mahdollisesti asetuksella on mahdollista muuttaa otsake, jota käyttäjä voi hyödyntää passiiviseen sormenjälkitunnisteeseen vaikuttaessa. Toisaalta `HTTP`-kutsun otsakkeissa voi olla eroja selaimien välillä, muun muassa otsakkeiden järjestyksessä, joten kokonaan toisen selaimen ilmoittaminen `User-Agent`-otsakkeessa voi olla käyttäjää jäljittävän tahon huomattavissa (Broenink, 2012).

Samat tiedot käyttöjärjestelmästä ja selaimesta löytyvät myös asiakaspuolelta JavaScriptin `navigator`-rajapinnasta. Käyttäjän voi vaihtaa `User-Agent`-otsakkeessa ilmoittamansa tiedot myös `navigator`-rajapintaan, jotta se ei eroa `HTTP`-kutsun ilmoittamista tiedoista. Ongelmana tässä on se, että JavaScriptillä on kuitenkin myös mahdollista tunnistaa selain ja sen versio myös muilla keinoin.

Selaimesta ja selaimen versiosta riippuen se sisältää tiettyjä selainmoottoriin ja JavaScript-moottoriin toteutettuja ominaisuuksia. Niiden välillä on eroavaisuuksia, josta ne pystytään tunnistamaan. Selaimen tunnistukseen voidaan käyttää esimerkiksi sen tukemien CSS-tyyliohjeiden tai JavaScript-toteutuksen ominaisuuksia. Koska selaimet kehittyvät jatkuvasti ja uudet versiot tuovat mukanaan uusia ominaisuuksia, voidaan niitä käyttämällä tunnistaa sekä selain, että sen versio. (Nikiforakis et al., 2013)

`User-Agent`-otsaketta ja asiakaspuolen vastaavia arvoja muuttamalla käyttäjä voi siis vaikuttaa ilmoitettaviin tietoihin, jolla voidaan pyrkiä esimerkiksi yhdenmukaistamaan sormenjälkitunnistetta isomman joukon käyttäjiä kanssa. Vääristettyjen tietojen, kuten eri selaimen tai selaimen version, ilmoittaminen asiakaspuolella voi kuitenkin olla käyttäjää jäljittävän tahon huomattavissa, ja näin yksilöidä käyttäjää yhä enemmän (Eckersley, 2010).

Tor Browser ilmoittaa selaimekseen sen pohjalla olevan Firefox-selaimen ja käyttöjärjestelmäkseen 'suositun Windows version' riippumatta käyttäjän oikeasta käyttöjärjestelmästä. Yhtenäisellä selainversiolla ja ilmoitetulla käyttöjärjestelmällä se pyrkii yhdenmukaistamaan `User-Agent`-otsakkeen ja asiakaspuolen vastaavat arvot Tor Browser -selaimen käyttäjien välillä, ja tätä kautta ehkäisemään sormenjälkitunnistamisen yksilöivyyttä. (Perry et al., 2013)

7.8 Välityspalvelinten käyttö

Käyttäjän IP-osoite sisältää tietoa käyttäjän palveluntarjoajasta ja maantieteellisestä sijainnista. Se voi myös käyttäjän Internet-liittymästä riippuen olla suurelta osin pysyvä, joten se on yksi yksilöivimpiä sormenjälkitunnisteen ominaisuuksista.

Verkkopalvelulle näkyvän IP-osoitteen vaihtaakseen käyttäjä voi reitittää verkkoliikenteensä toisen verkossa olevan laitteen kautta, eli välitys- tai VPN-palvelinta käyttäessä. Toinen laite on tällöin välityspalvelimena, ja verkkopalvelulle näkyvä IP-osoite on välityspalvelimen. (Nikiforakis et al., 2013)

Välityspalvelimen avulla voidaan käyttäjän oikean IP-osoitteen piilottamisen lisäksi esimerkiksi vaihtaa se vierailtavan sivuston perusteella tai tietyin väliajoin (Nikiforakis et al., 2013). Vaihtamalla IP-osoitetta useammin saadaan sormenjälkitunnistetta satunnaistettua. Toisaalta käyttäessä samaa välityspalvelinta useamman muun käyttäjän kanssa voidaan sormenjälkitunnistetta pyrkiä yhdenmukaistamaan.

Välityspalvelinta käyttäessä verkkopalvelulle näkyy käyttäjän laitteen IP-osoitteen sijaan välityspalvelimen osoite. Toisaalta myös TCP-liikenne toimii tällöin välityspalvelimen ohjelmiston avulla, joten verkkopalvelun on mahdollista tunnistaa muun muassa välityspalvelimen käyttöjärjestelmä (luku 6.5). Tämän mahdolliset eroavaisuudet HTTP-otsakkeisiin tai asiakaspuolen ominaisuuksiin voivat lisätä käyttäjän yksilöllisyyttä.

Myös muita verkkoliikenteen yksityisyyttä parantavia tekniikoita, kuten Tor-verkkoa voidaan käyttää tietoliikenteen sormenjälkitunnistamisen ehkäisemiseen muun muassa IP-osoitteen osalta.

7.9 Tor-verkko

Tor-verkko ja sipulireititys on alun perin Yhdysvaltain laivastossa ja DARPA:ssa (Defense Advanced Research Projects Agency) kehitetty tekniikka käyttäjän verkkoliikenteen anonymisointiin. Se käyttää sipulireititystä anonymisoimaan käyttäjän liikenteen ohjaamalla sen useamman välityspalvelimen kautta. Sipulireitityksessä kukin välityspalvelin tietää vain edellisen ja seuraavan välityspalvelimen osoitteen, sillä TCP-paketit ovat kerroksittain salattuja. Kukin välityspalvelin saa selville seuraavan välityspalvelimen osoitteen purkamalla sille asetetun salauskerroksen. Näin liikenteen alkuperä piilotetaan ulkopuolisilta, sillä koko välityspalvelimien kautta ohjatun reitin selvittäminen käyttäjään asti on tehty vaikeaksi.

(The Tor Project, 2010)

Käyttäjän verkkoliikenteen alkuperä on piilotettu myös verkkopalvelulta, sillä palvelu näkee vain viimeisen välityspalvelimen osoitteen. Välityspalvelimet myös satunnaistetaan ajoittain (The Tor Project, 2010). Verkkopalvelu ei siis saa selville käyttäjän `IP`-osoitetta. Verkkopalvelu voi kuitenkin selvittää, että käyttäjä käyttää Tor-verkkoa analysoimalla `TCP`-liikennettä tai vertaamalla `IP`-osoitetta tunnettujen Tor-välityspalvelimien listaan.

Toisaalta Tor-verkon ohjaamana välityspalvelimien kautta Internetin käyttö hidastuu jonkin verran. `HTTP`-kutsujen ja palvelimen vastauksien kulkiessa useamman maantieteellisesti laajasti sijoitettujen välityspalvelimen kautta niiden kiertomatka pidentyy (Dingledine et al., 2004).

Tor-verkon käyttö ei vaikuta evästeisiin ja muuhun selaimen tai laitteen sormenjälkitunnistamiseen, joten verkkoa ylläpitävä The Tor Project -järjestö kehittää myös sen ohella käytettäväksi suunniteltua Tor Browser -selainta. Tor Browserista kerron myöhemmin luvussa 7.12.

7.10 Yksityisen selauksen tila

Suurin osa selaimista tarjoaa mahdollisuuden käyttää yksityisen selauksen tilaa. Suurin osa toiminnallisuudesta liittyy käyttäjän jälkien peittämiseen hänen omalla koneellaan, mutta se voi auttaa myös jonkin verran käyttäjän seurantaan. Yksityisessä tilassa tallenteet ovat eristetty tavallisesta selaimen käytöstä, joten ne eivät seuraa käyttäjää yksityisen selauksen tilaan, tai toisin päin. Tallenteet on myös asetettu poistumaan kun yksityisen selauksen tilan ikkuna suljetaan. Firefoxin yksityisen selauksen tila sisältää myös ‘Tracking Protection’ -estolistan, joka estää tunnettujen käyttäjää seuraavien resurssien latautumisen. (Mozilla, 2015b)

7.11 Estolistat

Joidenkin selaimien asetuksilla tai lisäosilla on myös mahdollista estää tunnettujen käyttäjiä seuraavien verkkopalveluiden ja resurssien latautuminen selaimen toimesta. Kun toiminto huomaa selaimen hakevan tällaista resurssia, se peruuttaa kutsun lähetyksen, joten mahdolliset käyttäjän seurantaan käytettävät kuvat, JavaScript-ohjelmat tai muut resurssit eivät lataudu. Peruuttamalla kutsut resurssit eivät pääse asettamaan käyttäjän selaimeen tallenteita tai ajamaan mahdollisia yksityisyyttä loukkaavia skriptejä. (Kontaxis ja Chew, 2015)

Estolistat ovat ihmisten yhteistyönä keräämiä listoja käyttäjää seuraaviksi

havaituista URL- ja domain-osoitteista. Monet mainokset, varsinkin mainostajaverkkojen tarjoamista mainoksista, sisältävät käyttäjän seurantaa (Roesner et al., 2012), joten seurannan estolistoilla on myös hyvin paljon päällekkäisyyttä mainosten estolistojen kanssa. Firefoxissa sisäänrakennettu estolista on käytössä oletuksena yksityisen selauksen tilassa, mutta sen voi myös asettaa käyttöön normaaliin selaamiseen.

Tor Browser -selaimen kehittäjät päätyivät olemaan käyttämättä estolistoja perustellen sen estolistojen vaillinaisuudella. Koska estolistat toimivat mustan listan tavoin, on aina mahdollisuus, että osa käyttäjää seuraavista resursseista ei löydy listalta ja ohittaa eston. Estolistat voi myös rikkoa verkkosivuja estämällä tarpeellisten resurssien latautumisen. Sivustot saattavat myös yrittää tunnistaa estolistojen käyttöä, ja muuttaa sen mukaan sivuston toiminnallisuutta esimerkiksi estämällä sivun sisältöä. Tor Browser -selain pyrkii estämään käyttäjän seuraamisen muilla keinoin, josta kerron seuraavassa kappaleessa 7.12. (Perry et al., 2013)

7.12 Tor Browser

Tor Browser on Tor-verkon kehittäjien rinnakkainen projekti luoda yksityinen selain. Koska Tor-verkko itse keskittyy verkkoliikenteen, eli käyttäjän `TCP/IP`-tiedonsiirtoprotokollan ja samalla `IP`-osoitteen, anonymisointiin, aloittivat he kehittämään selainta, joka turvaisi käyttäjän anonymiteetin myös muilta käyttäjän seurantaa mahdollistamilta tekniikoilta. Tor Browser pyrkii estämään tallenteiden, kuten evästeiden, sekä sormenjälkitunnistuksen käytön käyttäjän seurantaan. (Perry et al., 2013)

Tor Browser listaa yksityisyyden päävaatimuksiin suojan tallenteiden käytöltä sivustojen väliseen seurantaan (Cross-Origin Identifier Unlinkability), suojan sormenjälkitunnisteen käytöltä sivustojen väliseen seurantaan (Cross-Origin Fingerprinting Unlinkability), sekä pitkäaikaisen seurannan torjunnan (Long-Term Unlinkability). Nämä tavoitteet selain pyrkii saavuttamaan monilla jo yllä mainituilla keinoilla, muun muassa eristämällä tallenteet, yhdenmukaistamalla sormenjälkitunnisteet, ja tarjoamalla helpon, osittain automatisoidun, tavan tallenteiden säännölliseen tyhjentämiseen. (Perry et al., 2013)

Tallenteita käytetään sivustojen väliseen seurantaan muun muassa asettamalla kolmannen osapuolen resursseilla evästeitä tai välimuistievästeitä. Tor Browser eristää tallenteet vieraillun sivuston mukaan, jolloin myös kolmannen osapuolen evästeet näkyvät vain siinä domain-osoitteessa, jolla käyttäjän vieraillessa eväs-

teet on asetettu. (luku 7.5)

Sormenjälkitunnisteen avulla käyttäjää voidaan mahdollisesti seurata sivustojen välisesti, sillä se pysyy samana sivustosta riippumatta. Tor Browser pyrkii vaikuttamaan sormenjälkitunnisteeseen väärentämällä selaimen ilmoittamia arvoja yhtenäisimmiksi, uudelleentoteuttamalla toiminnallisuuksia yksilöllisten tietojen piilottamiseksi, laitteiston ominaisuuksia virtualisoimalla, kysymällä hyväksyntää joillekin toiminnoille tai poistamalla niitä kokonaan käytöstä. (Perry et al., 2013)

Pitkäaikaista seurantaa verkkopalvelussa estääkseen Tor Browser pyrkii helpottamaan tallenteiden tyhjennystä selaimesta. Tämä toiminto löytyy selaimesta kontekstivalikon nappulasta 'New Identity' mikä tyhjentää kaikki selaimen tallenteet ja muutokset asetuksiin mitä käyttäjä on verkkosivuilla tehnyt. Selain myös oletusasetuksilla tyhjentää evästeet selaimen suljettua. (Perry et al., 2013)

Tor Browser perustuu Mozillan Firefox ESR -versioon, jonka toiminnallisuuksia on muutettu muutostiedostoilla ja oletusasetuksia vaihtamalla tai kiinnittämällä. Tor Browseriin kehitettyjä muutoksia on pyritty tuomaan myös itse Firefoxiin käytettäväksi projektissa 'Tor Uplift' (Steele, 2016). Muun muassa luvussa 7.5 mainittu tallenteiden eristäminen on alun perin kehitetty Tor Browser -selaimelle. Muutoksien avulla myös Firefox-selaimen käyttäjät voivat hyötyä käyttäjän seurantaa ehkäisevistä ominaisuuksista.

8 Pohdinta

Tallenteiden hallitseminen ja tyhjentäminen selaimissa on kehittynyt melko helppokäyttöiseksi. Evästeet, välimuisti ja muut tallenteet ovat tyhjennettävissä useimmiten muutamalla klikkauksella etsimällä selaimesta 'Poista historiatietoja...' -nappula. Useimmat selaimet eivät oletuksena automaattisesti tai ajoittaisesti poista evästeitä, joten käyttäjän tulee itse aktiivisesti tyhjentää tallenteita, tai valita sellaiset asetukset tai selaimen lisäosat millä ne poistuvat itsestään. Tallenteita tyhjentämällä käyttäjä voi vähentää verkkopalveluiden kerättävissä olevaa tietoa.

Siinä missä aikaisemmat sormenjälkitunnistamista mitanneet julkaisut ovat monesti havainneet aineistonsa sormenjälkitunnisteista yli 80% olevan yksilöllisiä, tuo Gómez-Boix et al. (2018) selkeästi yksityisyyden kannalta lupaavampia tuloksia mittaamalla sormenjälkitunnisteista noin 33% yksilöllisiksi. Lisäksi tutkimuksen anonymiteettijoukkojen koot ovat isompia kuin aikaisemmissa tutkimuksissa mitatut. Selaimen liitännäisten siirtyessä pois käytöstä niiden yksilöllistävä vaikutus tulee edelleen vähenemään. Flash-liitännäisen, joka on tällä hetkellä monissa selaimissa viimeinen tuettu liitännäinen, on määrä poistua käytöstä vuonna 2020 Adoben sille antaman tuen loputtua (Adobe, 2011). Varsinkin Tor, Mozilla ja Apple jatkavat selaimiensa sormenjälkitunnistusta ja käyttäjän seurantaa torjuvaa toiminnallisuutta.

Apple on kehittänyt tulevaan Safarin versio 12 selaimeensa 'Intelligent Tracking Prevention' -toimintoa torjumaan käyttäjän sivustojen välistä seurantaa tallenteiden avulla (Apple, 2018a). Ominaisuus eristää oletuksena kolmannen osapuolen evästeet toisistaan vieraillun sivun mukaan. Tapauksiin, joissa käyttäjä tarvitsee kolmannen osapuolen palveluun evästeet, tulee käyttäjän niiden käyttö erikseen hyväksyä. Apple mainitsi myös 2018 WWDC (Worldwide Developer Conference) -tilaisuudessa tulevansa rajoittamaan Safarin sormenjälkitunnisteiden yksilöllistävyyttä (Hautala, 2018).

Selainvalmistajien jatkaessa seurannan eston työkalujen kehitystä Internetin yksityinen käyttö saattaa tulla yhä helpommaksi. Seurannalta suojautuminen vaatii kuitenkin edelleen käyttäjältä oma-aloitteisuutta ja teknistä tietoa, sillä useimmat asetukset saattavat rikkoa joidenkin verkkopalveluiden toiminnallisuuksia. Rikkinäisen palvelun kohdatessaan käyttäjältä vaaditaan osaamista asetuksien tilapäiseen muuttamiseen, tai esimerkiksi vaihtoehtoisen selaimen käyttämiseen. Siksi yksityisyyteen liittyvät asetukset, kuten kolmannen osapuolen evästeiden esto tai Safarin 'Intelligent Tracking Prevention', ovat oletuksena pois päältä.

Internetin anonyymiin käyttöön pyrkivään Tor Browser -selaimeen on toteutettu eniten seurantaa torjuvia parannuksia, ja ne ovat oletuksena päällä. Se sopii käyttäjälle, joka on valmis parantamaan yksityisyyttään selainten ominaisuuksien ja verkkopalveluiden toimintavarmuuden kustannuksella. Tor Browser käyttää myös rinnakkaisprojekti Tor-verkkoa, joka on suunniteltu anonymisoimaan itse TCP/IP-liikenne, jonka myötä Internetin käyttö hidastuu jonkin verran liikenteen ohjauduttua useamman välityspalvelimen kautta. Toisaalta joitakin Tor Browser -selaimeen alunperin kehitettyjä yksityisyyttä parantavia ominaisuuksia siirretään pikkuhiljaa myös Firefox-selaimeen, mahdollistaen niiden käytön ottamatta välttämättä koko Tor Browseria käyttöön.

Viiteluettelo

Acar, G., Eubank, C., Englehardt, S., Juarez, M., Narayanan, A., and Diaz, C. (2014). The Web Never Forgets: Persistent Tracking Mechanisms in the Wild. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 674–689, New York, NY, USA. ACM.

Acar, G., Juarez, M., Nikiforakis, N., Diaz, C., Gürses, S., Piessens, F., and Preneel, B. (2013). FPDetective: Dusting the Web for Fingerprinters. In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 1129–1140, New York, NY, USA. ACM.

Adobe (2011). On Improving Privacy: Managing Local Storage in Flash Player. https://blogs.adobe.com/digitalmedia/2011/01/on-improving-privacy-managing-local-storage-in-flash-player/. [Last accessed 10-Nov-2017].

Adobe (2017). Flash & The Future of Interactive Content. https://theblog.adobe.com/adobe-flash-update/. [Last accessed 02-Jul-2018].

Alaca, F. and van Oorschot, P. C. (2016). Device fingerprinting for augmenting web authentication: classification and analysis of methods. In: *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pages 289–301. ACM.

Apple (2018a). Intelligent Tracking Prevention 2.0. https://webkit.org/blog/8311/intelligent-tracking-prevention-2-0/. [Last accessed 25-Jul-2018].

Apple (2018b). Safari 12. https://developer.apple.com/safari/whats-new/. [Last accessed 18-Jun-2018].

Ayenson, M. D., Wambach, D. J., Soltani, A., Good, N., and Hoofnagle, C. J. (2011). Flash cookies and privacy II: Now with HTML5 and ETag respawning.

Boda, K., Földes, A. M., Gulyás, G. G., and Imre, S. (2012). User Tracking on the Web via Cross-browser Fingerprinting. In: *Proceedings of the 16th Nordic Conference on Information Security Technology for Applications*, NordSec'11, pages 31–46, Berlin, Heidelberg. Springer-Verlag.

Broenink, R. (2012). Using browser properties for fingerprinting purposes. In: *Proceedings of the 16th Biennial Twente Student Conference on IT, Enschede, the Netherlands*.

BuiltWith (2018). Facebook CDN Usage Statistics. https://trends.builtwith.com/cdn/Facebook-CDN. [Last accessed 04-Jul-2018].

Cao, Y., Li, S., and Wijmans, E. (2017). (Cross-)Browser Fingerprinting via OS and Hardware Level Features. In: *Proceedings of the Network & Distributed System Security Symposium*. NDSS.

Castelluccia, C., Olejnik, L., and Minh-Dung, T. (2014). Selling off privacy at auction. In: *Proceedings of Network and Distributed System Security Symposium*. NDSS.

Deutschmann, I. and Lindholm, J. (2013). Behavioral biometrics for DARPA's active authentication program. In: *Biometrics Special Interest Group (BIOSIG), 2013 International Conference of the*, pages 1–8. IEEE.

Dingledine, R., Mathewson, N., and Syverson, P. (2004). Tor: The Second-Generation Onion Router. In: *Proceedings of the 13th USENIX Security Symposium*, pages 303–320. USENIX.

Do, K. (2017). Cookie AutoDelete. https://github.com/Cookie-AutoDelete/Cookie-AutoDelete/wiki/FAQ:-Common-Questions-and-Issues/0c0fc45071afce8cad0048be26d74b680d129052. [Last accessed 03-Jul-2018].

Eckersley, P. (2010). How unique is your web browser? In: *International Symposium on Privacy Enhancing Technologies Symposium*, pages 1–18. Springer.

Englehardt, S. and Narayanan, A. (2016). Online Tracking: A 1-million-site Measurement and Analysis. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 1388–1401, New York, NY, USA. ACM.

Fifield, D. and Egelman, S. (2015). Fingerprinting web users through font metrics. In: *International Conference on Financial Cryptography and Data Security*, pages 107–124. Springer.

Gómez-Boix, A., Laperdrix, P., and Baudry, B. (2018). Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale. In: *Proceedings of the 2018 World Wide Web Conference*, pages 1–10, Lyon, France.

Google (2013). Share Chrome with others. https://support.google.com/chrome/answer/2364824. [Last accessed 03-Jul-2018].

Google (2016). Avoiding the Not Secure Warning in Chrome. https://developers.google.com/web/updates/2016/10/avoid-not-secure-warn. [Last accessed 3-Nov-2017].

Google (2018). A secure web is here to stay. https://security.googleblog.com/2018/02/a-secure-web-is-here-to-stay.html. [Last accessed 03-Mar-2018].

Gulyás, G., Schulcz, R., and Imre, S. (2008). Comprehensive analysis of web privacy and anonymous web browsers: are next generation services based on collaborative filtering? In: *Joint SPACE and TIME International Workshops*.

Hautala, L. (2018). Here's how Apple's browsing privacy features will work in Safari. https://www.cnet.com/news/heres-how-apples-browsing-privacy-features-will-work-safari/. [Last accessed 25-Jul-2018].

IETF (2014). Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests. https://tools.ietf.org/html/rfc7232. [Last accessed 4-Nov-2017].

Jagpal, N., Dingle, E., Gravel, J.-P., Mavrommatis, P., Provos, N., Rajab, M. A., and Thomas, K. (2015). Trends and Lessons from Three Years Fighting Malicious Extensions. In: *USENIX Security Symposium*, pages 579–593.

Järvinen, P. (2010). *Yksityisyys*. Docendo.

Juels, A., Jakobsson, M., and Jagatic, T. N. (2006). Cache cookies for browser authentication. In: *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 5–pp. IEEE.

Kapravelos, A., Grier, C., Chachra, N., Kruegel, C., Vigna, G., and Paxson, V. (2014). Hulk: Eliciting Malicious Behavior in Browser Extensions. In: *Proceedings of the 23rd USENIX Security Symposium*, pages 641–654. USENIX.

Kolšek, M. (2002). Session fixation vulnerability in web-based applications. http://www.acros.si/papers/session_fixation.pdf. [Last accessed 4-April-2018].

Kontaxis, G. and Chew, M. (2015). Tracking Protection in Firefox For Privacy and Performance. In: *Proceedings of the 9th Workshop on Web 2.0 Security and Privacy (W2SP)*.

Krishnamurthy, B., Naryshkin, K., and Wills, C. (2011). Privacy leakage vs. protection measures: the growing disconnect. In: *Proceedings of the Web*, volume 2, pages 1–10.

Laperdrix, P., Rudametkin, W., and Baudry, B. (2016). Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. In: *Security and Privacy (SP), 2016 IEEE Symposium on*, pages 878–894. IEEE.

Leon, P., Ur, B., Shay, R., Wang, Y., Balebako, R., and Cranor, L. (2012). Why Johnny can't opt out: a usability evaluation of tools to limit online behavioral advertising. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 589–598. ACM.

MaxMind (2002). IP Geolocation and Online Fraud Prevention. https://www.maxmind.com. [Last accessed 26-May-2018].

Mayer, J. R. and Mitchell, J. C. (2012). Third-Party Web Tracking: Policy and Technology. In: *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, pages 413–427.

Microsoft (2016). Microsoft Privacy Statement. https://privacy.microsoft.com/en-us/privacystatement/. [Last accessed 28-Jun-2018].

Mozilla (2012a). Disable third-party cookies in Firefox to stop some types of tracking by advertisers. https://support.mozilla.org/en-US/kb/disable-third-party-cookies. [Last accessed 30-Jun-2018].

Mozilla (2012b). Enable and disable cookies that websites use to track your preferences. https://support.mozilla.org/en-US/kb/enable-and-disable-cookies-website-preferences. [Last accessed 30-Jun-2018].

Mozilla (2014a). Basic concepts. https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API/Basic_Concepts_Behind_IndexedDB. [Last accessed 10-May-2018].

Mozilla (2014b). Canvas API. https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API. [Last accessed 1-Jun-2018].

Mozilla (2014c). Cookies. https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies. [Last accessed 29-Oct-2017].

Mozilla (2014d). Delete browsing, search and download history on Firefox. https://support.mozilla.org/en-US/kb/delete-browsing-search-download-history-firefox. [Last accessed 04-Jul-2018].

Mozilla (2014e). Web Storage API. https://developer.mozilla.org/en-US/docs/Web/API/Web_Storage_API. [Last accessed 06-Oct-2017].

Mozilla (2015a). Etag. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/ETag. [Last accessed 03-Oct-2017].

Mozilla (2015b). Private Browsing - Use Firefox without saving history. https://support.mozilla.org/en-US/kb/private-browsing-use-firefox-without-history. [Last accessed 01-Jul-2018].

Mozilla (2016). HTTP caching. https://developer.mozilla.org/en-US/docs/Web/HTTP/Caching. [Last accessed 04-Oct-2017].

Mozilla (2017a). 52.0 Firefox Release. https://www.mozilla.org/en-US/firefox/52.0/releasenotes/. [Last accessed 18-Jun-2018].

Mozilla (2017b). privacy.websites. https://developer.mozilla.org/en-US/Add-ons/WebExtensions/API/privacy/websites. [Last accessed 30-Jun-2018].

Mozilla (2018). Facebook Container Extension: Take control of how you're being tracked. https://blog.mozilla.org/firefox/facebook-container-extension/. [Last accessed 04-Jul-2018].

Nikiforakis, N., Joosen, W., and Livshits, B. (2015). Privaricator: Deceiving fingerprinters with little white lies. In: *Proceedings of the 24th International Conference on World Wide Web*, pages 820–830. International World Wide Web Conferences Steering Committee.

Nikiforakis, N., Kapravelos, A., Joosen, W., Kruegel, C., Piessens, F., and Vigna, G. (2013). Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting. In: *2013 IEEE Symposium on Security and Privacy*, pages 541–555. IEEE SP.

OWASP (2006a). Cross-Site Request Forgery (CSRF). https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF). [Last accessed 2-Nov-2017].

OWASP (2006b). Cross Site Scripting Flaw. https://www.owasp.org/index.php/Cross-site_Scripting_(XSS). [Last accessed 2-Nov-2017].

Perry, M., Clark, E., Murdoch, S., and Koppen, G. (2013). The Design and Implementation of the Tor Browser [DRAFT]. https://www.torproject.org/projects/torbrowser/design/. [Last accessed 30-Jun-2018].

Projects, T. C. (2018). Flash Usage Trends. https://www.chromium.org/flash-roadmap/flash-usage-trends. [Last accessed 20-Nov-2018].

Roesner, F., Kohno, T., and Wetherall, D. (2012). Detecting and Defending Against Third-party Tracking on the Web. In: *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 12–12, Berkeley, CA, USA. USENIX Association.

Schmucker, N. (2011). Web tracking, SNET2 Seminar Paper-Summer Term.

Silvennoinen, P. (2004). Yksityisyys ja henkilötiedot www-sivustoilla. Pro gradu, Tampereen yliopisto.

Sood, A. and Enbody, R. (2011). Malvertising - Exploiting web advertising. *Computer Fraud and Security*, 2011(4):11–16.

Sorensen, O. (2013). Zombie-cookies: Case studies and mitigation. In: *Proceedings of the 8th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 321–326. IEEE.

Steele, S. (2016). Tor at the Heart: Firefox. https://blog.torproject.org/tor-heart-firefox. [Last accessed 01-Jul-2018].

The Tor Project (2010). Tor: Overview. https://www.torproject.org/about/overview.html.en. [Last accessed 20-Jul-2018].

Thomas, K., Bursztein, E., Grier, C., Ho, G., Jagpal, N., Kapravelos, A., McCoy, D., Nappa, A., Paxson, V., Pearce, P., et al. (2015). Ad injection at scale: Assessing deceptive advertisement modifications. In: *Proceedings of the 2015 IEEE Symposium on Security and Privacy (SP)*, pages 151–167. IEEE.

Upturn (2016). What ISPs Can See. https://ecfsapi.fcc.gov/file/60002077347.pdf. [Last accessed 3-Nov-2017].

Vastel, A., Laperdrix, P., Rudametkin, W., and Rouvoy, R. (2018). P-STALKER: Tracking Browser Fingerprint Evolutions. In: *Proceedings of the 39th IEEE Symposium on Security and Privacy*, pages 1–14. IEEE SP.

Weiss, A., Ramapanicker, A., Shah, P., Noble, S., and Immohr, L. (2007). Mouse movements biometric identification: A feasibility study.

Yen, T.-F., Xie, Y., Yu, F., Yu, R. P., and Abadi, M. (2012). Host Fingerprinting and Tracking on the Web:Privacy and Security Implications. In: *Proceedings of the 19th Annual Network and Distributed System Security Symposium*. NDSS.

Zalewski, M. (2012). p0f v3: passive fingerprinter. http://lcamtuf.coredump.cx/p0f3/README. [Last accessed 02-Jul-2018].

A Zombieväste-esimerkki

```
const express = require("express");
const bodyParser = require("body-parser");
const cookieParser = require("cookie-parser");
const path = require("path");

/**
 * Key-value object to store and map fingerprints to identifiers.
 * @type {object}
 */
const FINGERPRINTS = {};

/**
 * Page header.
 * @type {string}
 */
const HEADER = `
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="public/style.css" type="text/css"/>
<script type="text/javascript" src="/public/fingerprint2.js"></script>
<script type="text/javascript">
function sendFingerprint() {
    let req = new XMLHttpRequest();

    req.addEventListener("load", function(event) {
        console.log(event);

        let nickname = event.target.responseText;
        if (!nickname) {
            console.log("Fingerprint not found.");
            return;
        }
        console.log("Received nickname", nickname);

        let title = document.getElementById("title");
        let correspondingName = document.getElementById("corresponding-nickname");
        title.innerHTML = "Your nickname: " + nickname;
        correspondingName.innerHTML = nickname;
    });
    new Fingerprint2().get(function(fingerprint) {
        let fingerprintElement = document.getElementById("fingerprint");
        fingerprintElement.innerHTML = fingerprint;
```

```
        console.log("Sending fingerprint", fingerprint);
        req.open("POST", "/cookie-refreshing/fp");
        req.send(fingerprint);
    });
}
</script>
</head>
<body>
<main>
`;

/**
 * Page footer.
 * @type {string}
 */
const FOOTER = `
</main>
</body>
</html>
`;

/**
 * HTTP endpoint for home page.
 * @param req HTTP request object
 * @param res HTTP response object
 */
function home(req, res) {
    const html = `
${HEADER}
<h2>Cookie respawning</h2>
<p>Fill in a nickname you wish to use. Valid characters are a-Z and 0-9.
Your nickname will be stored in your browser cookies and cache.</p>
<form method="POST" action="/">
  <input type="text" name="nickname" required="required" pattern="[a-zA-Z0-9]+"/><br/>
  <input type="submit" name="cookie-refreshing" value="Try cookie refreshing">
</form>
${FOOTER}
  `;
    res.send(html);
}

/**
 * HTTP endpoint for POST / form submission.
 * @param req request object
```

```
 * @param res request object
 */
function setCookie(req, res) {
    let redirect = '';
    if (req.body["cookie-syncing"]) {
        redirect = "/cookie-syncing";
    } else if (req.body["cookie-refreshing"]) {
        redirect = "/cookie-refreshing";
    }

    const nickname = req.body["nickname"];
    if (!nickname || !nickname.match("^[a-zA-Z0-9]+$")) {
        res.send(400);
        return;
    }
    res.cookie("nickname", nickname, {
        path: redirect,
        expires: new Date(Date.now() + 1000 * 60 * 60 * 24 * 2)
    });
    res.redirect(302, redirect);
}

/**
 * Create HTML for /cookie-refreshing page.
 * @param nickname Current recognized nickname
 * @param cookie   Current cookie value
 * @param etag     Current ETag value
 * @returns {string}
 */
const cookieRefreshingHTML = (nickname, cookie, etag) => `
${HEADER}
<h2 id="title">Your nickname: ${nickname}</h2>
<table>
  <tr>
    <td>Cookie</td>
    <td id="cookie-value">${cookie}</td>
  </tr>
  <tr>
    <td>ETag</td>
    <td id="etag-value">${etag}</td>
  </tr>
  <tr>
    <td>Browser fingerprint</td>
    <td id="fingerprint">...</td>
```

```
  </tr>
  <tr>
    <td>Name via fingerprint</td>
    <td id="corresponding-nickname">...</td>
  </tr>
</table>
<script type="text/javascript">
sendFingerprint();
</script>
${FOOTER}
`;

/**
 * HTTP endpoint for /cookie-refreshing page.
 * @param req HTTP request object
 * @param res HTTP response object
 */
function cookieRefreshing(req, res) {
    const cookie = req.cookies["nickname"]; // Get nickname from cookie
    const etag = req.get("If-None-Match"); // Get nickname from ETag header
    if (cookie || etag) {
        res.cookie("nickname", cookie || etag, {
            path: "/cookie-refreshing",
            expires: new Date(Date.now() + 1000 * 60 * 60 * 24)
        }); // Refresh cookie.
        res.set("ETag", cookie || etag); // Refresh ETag.
    }
    const response = cookieRefreshingHTML(
        cookie || etag || "Unknown",
        cookie || "...",
        etag || "..."
    );
    res.send(response);
}

/**
 * AJAX endpoint for receiving computed browser fingerprint.
 * @param req HTTP request object
 * @param res HTTP response object
 */
function cookieRefreshingReceiveFingerprint(req, res) {
    // Fingerprint received from user in AJAX request body.
    const fingerprint = req.body;
    if (!fingerprint) {
```

```
        res.status(200).end();
        return;
    }
    const found_nickname = FINGERPRINTS[fingerprint];
    // If fingerprint found from the database use it to refresh cookie.
    if (found_nickname) {
        console.log(`Fingerprint found for ${found_nickname}.`);
        res.cookie("nickname", found_nickname, {
            path: "/cookie-refreshing",
            expires: new Date(Date.now() + 1000 * 60 * 60 * 24)
        });
        res.send(found_nickname);
        return;
    }
    const nickname = req.cookies["nickname"];
    // If cookies contain identifier map it to received fingerprint.
    if (nickname) {
        console.log(`Setting fingerprint ${fingerprint} for ${nickname}.`);
        FINGERPRINTS[fingerprint] = nickname;
    }
    res.status(200).end();
}

const app = express();
app.get("/", home);
app.post("/", bodyParser.urlencoded({extended: false}), setCookie);
app.get("/cookie-refreshing", cookieParser(), cookieRefreshing);
app.post(
    "/cookie-refreshing/fp",
    bodyParser.text(),
    cookieParser(),
    cookieRefreshingReceiveFingerprint
);
app.use("/public", express.static(path.join(__dirname, "public")));
app.set("etag", false);
/* Disable Express server's default ETag caching. */
Object.defineProperty(app.request, "fresh", {
    configurable: true,
    enumerable: true,
    get: () => false
});

/* Start app. */
const PORT = 80;
```

```
app.listen(PORT, () => {
    console.log("Starting server on port " + PORT);
});
```